RITA Intelligent Transportation Systems Joint Program Office

Core System System Architecture Document (SAD)

www.its.dot.gov/index.htm

Initial Draft — June 13, 2011

U.S. Department of Transportation

Research and Innovative Technology Administration

Produced by Lockheed Martin
ITS Joint Program Office
Research and Innovative Technology Administration
U.S. Department of Transportation

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE (DD MM YYYY) 13 06 2011	**2. REPORT TYPE** (Draft Specification)	**3. DATES COVERED** Month YYYY – Month YYYY

4. TITLE AND SUBTITLE **Core System: System Architecture Document (SAD) Initial Draft**	**5a. CONTRACT NUMBER** GS-23F-0150S
	5b. GRANT NUMBER Xxxx
6. AUTHOR(S) Core System Engineering Team	**5c. PROGRAM ELEMENT NUMBER** Xxxx
	5d. PROJECT NUMBER DTFH61-10-F-00045
	5e. TASK NUMBER Xxxx
	5f. WORK UNIT NUMBER Xxxx
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Lockheed Martin 9500 Godwin Drive Manassas, VA 20110	**8. PERFORMING ORGANIZATION REPORT NUMBER** 11-USDOTSE-LMDM-00024
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) US Department of Transportation Research and Innovative Technology Administration ITS Joint Program Office 1200 New Jersey Ave., S.E. Washington D.C. 20590	**10. SPONSORING/MONITOR'S ACRONYM(S)** Xxxx
	11. SPONSORING/MONITOR'S REPORT NUMBER(S) Xxxx
12a. DISTRIBUTION/AVAILABILITY STATEMENT This document is available to the public through the National Technical Information Service, Springfield, Virginia 22161.	**12b. DISTRIBUTION CODE** Xxxx
13. SUPPLEMENTARY NOTES Xxxx	

14. ABSTRACT (Maximum 200 words)
Xxxx

15. SUBJECT TERMS
Xxxx

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT** None	**18. NUMBER OF PAGES** Xxxx	**19a. NAME OF RESPONSIBLE PERSON** Walt Fehr
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified			**19b. TELEPHONE NUMBER** (202) 366-0278

CHANGE LOG

Revision	*Change Summary*	*Author*	*Date*
-	Initial Release	Lockheed Martin	6/13/2011

READER'S GUIDE

This System Architecture Document is a work in progress. The initial draft is focused on Enterprise, Functional and Connectivity viewpoints, and includes several alternatives for different aspects of the Core System from the perspective of those architecture viewpoints. The reader is encouraged to study the viewpoint specifications in section 3, and then the associated views in section 4 with emphasis on those views for which alternatives are provided. The introductory material in Section 4 includes a listing of views that include alternatives.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1.0 INTRODUCTION

1.1 Identification

This document is the System Architecture Document (SAD) for the Core System for the United States Department of Transportation's (USDOT) *connected vehicle* program.

1.2 Document Overview

The USDOT initiated this Systems Engineering (SE) project to define the Concept of Operations (Con-Ops), requirements, and architecture for the Core System that will enable safety, mobility, and environmental applications in an environment where vehicles and personal mobile devices interact wirelessly, hereafter referred to as the connected vehicle environment.

The intended use of this SAD is to describe the system architecture of the Core System by following Standard 1471-2000 of the Institute of Electrical and Electronics Engineers (IEEE), the IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.

The IEEE 1471-2000 standard suggests the use of views for documenting different aspects of a software intensive system, without recommending particular viewpoints. The standard does discuss several example viewpoints:

- Structural and Behavioral Viewpoints, which are used for software-intensive systems,
- Physical Interconnect and Link Bit Error Rate Viewpoints, which focus on communications and one particular performance metric (bit error rate),
- Decomposition and Allocation Viewpoint, which incorporates requirements development and traceability to components of the system architecture,
- Enterprise, Information, Computational, Engineering and Technology Viewpoints of the Reference Model of Open Distributed Processing (RM-ODP), defined by ISO/IEC 10746-3: 1996. These viewpoints provide an architectural framework suited to distributed processing systems, in particular those "in which discrete components may be located in different places, or where communications between components may suffer delay or may fail."

This last model is most appropriate to the Core System and how it interacts with the connected vehicle environment. Functionality within the connected vehicle environment is distributed between mobile and non-mobile entities and the Core, and interactions take place over a variety of wired and wireless links with varying performance characteristics subject to environmental degradation and failure.

The Consultative Committee for Space Data Systems (CCSDS) extended the RM-ODP framework and refined the five viewpoints in the Reference Architecture for Space Data Systems (RASDS). These Viewpoints include most concepts of the RM-ODP Viewpoints but are better suited to identifying layered interfaces.

Some of the elements that the Core System interacts with are constantly in motion and may only occasionally be in contact with other elements and the Core. The *connected vehicle* environment is dependent on wireless communications that requires functionality at all communications layers to address security concerns within an intermittently connected system. The RASDS identifies Viewpoints that are more applicable to address these dependencies and concerns. Consequently this architecture description uses the RASDS framework.

Within RASDS, the Enterprise and Information viewpoints are similar to viewpoints of the same name in RM-ODP. RM-ODP's Computational viewpoint is replaced with the Functional viewpoint, which addresses the same concerns but uses slightly different language. The Engineering and Technology viewpoints are replaced with the Connectivity and Communications viewpoints. The replacements include most of the information from the RM-ODP counterparts but have a greater focus on the disconnected nature of interacting elements, which makes these views preferable for the Core System.

The System Architecture document consists of the following sections:

- Section 1.0 provides an introduction to this document.
- Section 2.0 identifies the stakeholders and their concerns that were considered when developing the system architecture.
- Section 3.0 introduces the concepts of architecture viewpoints and views, and further details the viewpoints that have been selected to address stakeholder concerns.
- Section 4.0 presents detailed architecture views for each of the viewpoints introduced in Section 3.0.
- Section 5.0 describes the known inconsistencies among the architecture views as well as a discussion of the consistencies across all views.
- Section 6.0 details the rational for the architecture views selected.
- Section 7.0 contains the appendix which has a table summarizing the contents of the architecture viewpoints.
- Section 8.0 contains the glossary and acronym list

1.3 Scope

This document presents the System Architecture of the Core System. This section will be expanded later to summarize the services and interfaces included in the Core System architecture.

1.4 Context

Figure 1-1 shows the context in which the Core System resides as documented in the Concept of Operations (ConOps).

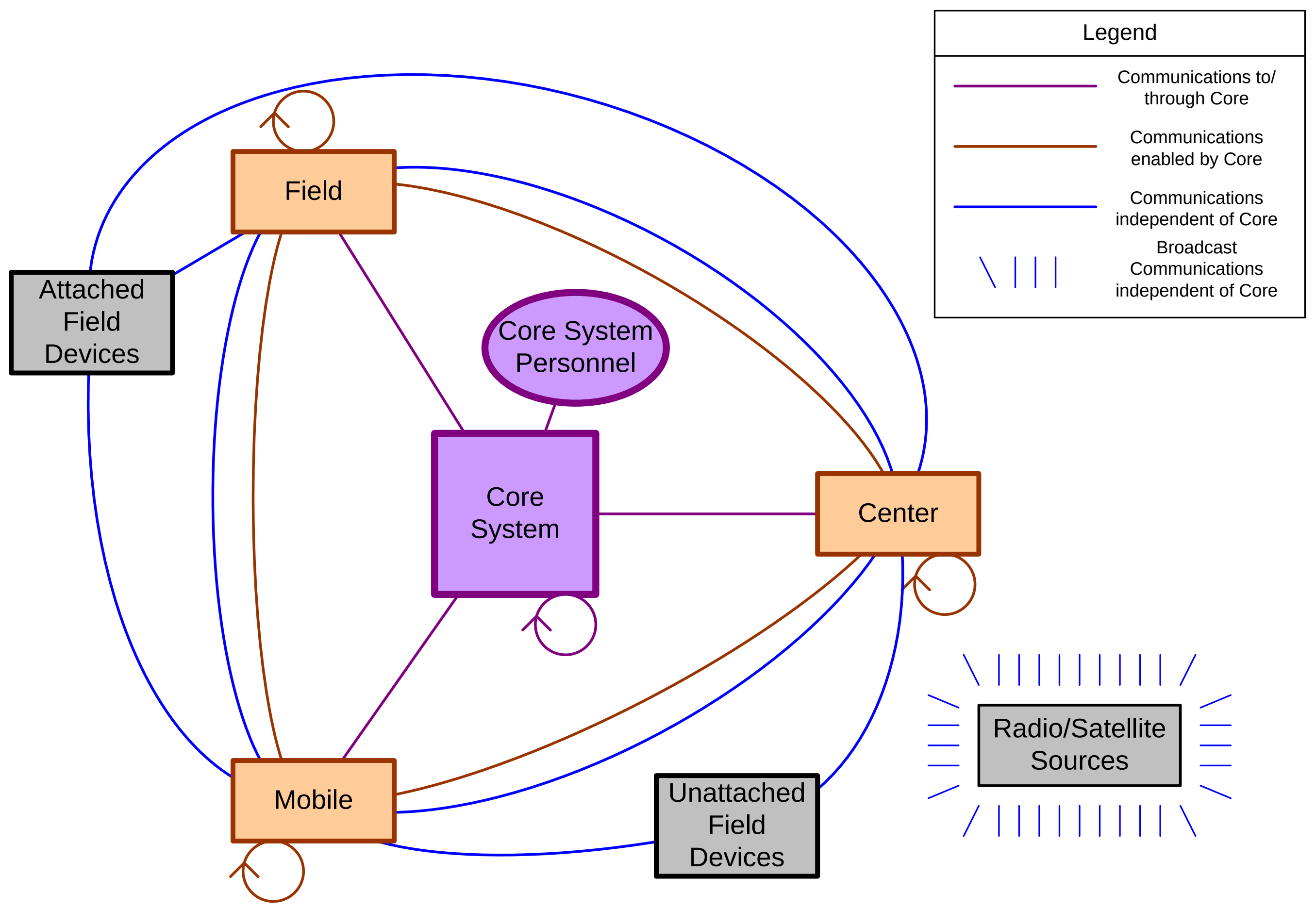

Legend
Communications to/ through Core
Communications enabled by Core
Communications independent of Core
Broadcast Communications independent of Core
Field
Attached Field Devices
Core System Personnel
Core System
Center
Mobile
Unattached Field Devices
Radio/Satellite Sources

Figure 1-1: Core System Context Diagram

The Core System interacts with four types of entities:

- **Mobile** includes all vehicle types (private/personal, trucks, transit, emergency, commercial, maintenance, and construction vehicles) as well as non-vehicle-based platforms including portable personal devices (smart phones, tablets, etc.) used by travelers (vehicle operators, passengers, cyclists, pedestrians, etc.) to provide and receive transportation information. Mobile interacts with other Mobile and Field entities (e.g. Dynamic Message Signs (DMS), Roadside Equipment (RSE)) in Mobile's vicinity, and Center from any location.
- **Field** represents the intelligent infrastructure distributed near or along the transportation network which perform surveillance (e.g. traffic detectors, cameras), traffic control (e.g. signal controllers), information provision (e.g. DMS) and local transaction (e.g., tolling, parking) functions. Typically, their operation is governed by transportation management functions running in back offices. Field also includes RSE supporting Dedicated Short Range Communications (DSRC), and other non-DSRC wireless communications infrastructure that provides communications between Mobile elements and fixed infrastructure[1].

[1] Technically, DSRC between RSEs and mobile elements are part of the communications layer. The placement of the RSE with respect to the Core System is called out specifically; however, in the past the RSE has typically been considered part of "the system".

- **Center** represents back office systems including public and commercial transportation and non-transportation systems that provide management, administrative, information dissemination, and support functions. These systems may communicate with other center systems to enable coordination between modes of transportation (e.g. transit and traffic management) and across jurisdictions. All of these systems take advantage of the Core System to provide or also make use of application data.
- **Core System Personnel** represents the people that operate and maintain the Core System. In addition to network managers and operations personnel, Core System Personnel also includes those that are deploying and provisioning Core elements. Other personnel interacting with the Core include developers of software services that, maintain, fix and expand Core services or extend the system as required.

1.5 References

Documents listed below were used in the preparation of this SAD.

- Core System Concept of Operations, 19 April 2011.
- Core System System Requirements Specification (SyRS), 18 April 2011.
- CAMP Security Final Report, Draft, 8/31/2010
- CCSDS 311.0-M-1 – Reference Architecture for Space Data Systems, Recommended Practice, September 2008
- IEEE Std. 1471 – IEEE Recommended Practice for Architectural Description of Software Intensive Systems, 21 September 2000.
- IEEE 1609.2 Standard for Wireless Access in Vehicular Environments (WAVE) - Security Services for Applications and Management Messages, Jun 2006
- IEEE 1609.3 Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services, Apr 2007
- OMG Systems Modeling Language (OMG SysML^{TM}), version 1.1, November 2008.
- SAE J2735 - Dedicated Short Range Communications (DSRC) Message Set Dictionary, v2, Nov 2009
- VII Privacy Policies Framework, Version 1.0.2, February 16, 2007.

2.0 STAKEHOLDERS AND CONCERNS

A system stakeholder is an individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system. Concerns are those interests which pertain to the system's development, its operation or any other aspects that are critical or otherwise important to one or more stakeholders[2]. All needs and goals and most constraints identified in the ConOps are reflected by one or more concerns, as documented in the description of those concerns.

2.1 Stakeholders

In the following sections the different stakeholder roles relevant to the Core System are described. IEEE 1471-2000 requires that at least the following stakeholders be considered: Users, Acquirers, Developers, and Maintainers. Users are subdivided because some have different concerns, and because their operating environments are all substantially different.

2.1.1 Users

The Concept of Operations defines three System User types: Mobile, Field and Center. These are devices that interact with the Core System. Each of these devices is associated with one or more stakeholders, including those that create, maintain and use them. The User stakeholder is the operator of the Mobile, Field, or Center device. These three System User types all fall under the IEEE concept of "user", but owing to the significant logistical differences between them, they are identified individually. Operators are a fourth user type—not a System User, but a user nonetheless in that Operators are people that interact with the Core.

2.1.1.1 Mobile User

The Mobile User operates a personal information device (e.g. smart phone) or operates or rides in a vehicle that uses Core Services. Mobile Users include automobile, bus, truck, construction and emergency vehicle drivers and passengers, as well as cyclists and pedestrians.

2.1.1.2 Field User

The Field User owns, operates and/or maintains the intelligent infrastructure distributed near or along the transportation network which performs surveillance, information provision, local transactions, and control functions.

2.1.1.3 Center User

The Center User owns, operates, and/or maintains back office systems that use Core Services. Center User stakeholders include public and commercial transportation, transit and fleet managers, vehicle and device manufacturers, information service providers, emissions, commercial vehicle and other regulatory managers, toll administrators, maintenance and construction operators and transportation data aggregators.

2.1.1.4 Operator

The Operator is the day-to-day administrator of Core System services. The Operator is the user with authority to control delivery of services and manage interactions between Cores. The Operator has responsibility for system configuration and any manual processes that are part of day-to-day Core operations.

[2] See IEEE 1471-2000.

The Operator role will usually be funded by the same entity that serves as the Manager. The Operator may be part of the same agency as the Manager, or it may be contracted to a separate entity under control of the Manager.

2.1.2 Acquirer/Deployer

The acquirer is the entity that procures a Core System. .

Public entities that may assume the Acquirer role include state Departments of Transportation (DOTs), transit agencies, commercial vehicle administration agencies, or regional coordination entities that include elements of multiple such agencies. Private entities that may assume the Acquirer role include commercial fleet management, vehicle manufacturer or information service provider (e.g., traveler information) entities.

2.1.3 Maintainer/Administrator

The Maintainer is the role that ensures continuous system operation by ensuring the availability of resources required for the Core System to operate properly.

The Maintainer role will usually be funded by the same entity that serves as the Manager. The Maintainer may be part of the same agency as the Manager, or it may be contracted to a separate entity under control of the Manager.

2.1.4 Developer

The Developer is responsible for transforming the system architecture into a functioning system that meets the requirements set forth in the System Requirements Specification.

The Developer role will likely be assumed by commercial software development and hosting companies.

2.1.5 Manager

The Manager is responsible for planning the deployment and managing the operations of the Core System. The Manager does not interact directly with the Core. Changes to the Core that require operational modification (to enforce policies for example) are determined by the Manager, but implemented by the Operator.

The Manager role will be part of an agency or it may be contracted to a separate entity under control of an agency.

2.1.6 Tester

The Tester is responsible for verifying that changes made to Core System functionality are properly implemented. This includes both validation and verification steps.

2.1.7 Policy-Setter

The Policy-Setter determines policies that affect Core System deployment, implementation or operations. This could be a local entity such as a DoT, a regional body or a federal entity such as the FCC.

2.2 Concerns of Stakeholders

Concerns are concepts that are of interest or importance to one or more stakeholders. Concerns may apply to any phase of the system life cycle. However, some concerns may apply more during system design, or implementation, or operations, and not during other phases of the system’s life.

2.2.1 Performance

This concern deals with characteristics such as speed (responsiveness of Core System services) and availability, reliability, capacity and other quantitative measures.

2.2.2 Interfaces

This concern deals with how interfaces are defined and how users access and operate with the Core System.

2.2.3 Functionality

This concern deals with how the Core System functions internally; how its components work together and how they transition between operational modes.

2.2.4 Security

This concern deals with the security of Core System services, in terms of maintaining sole control, system integrity and integrity of the information passing to and through the Core.

2.2.5 Organization/Resources

This concern is about organization of system development, resources required to develop, deploy, maintain and operate the system.

2.2.6 Appropriateness

This concern asks whether the delivered Core System fulfills the needs set forth in the Concept of Operations and meets the overall goals defined therein.

2.2.7 Feasibility

This concern deals with feasibility, with regard to state of current technology, deployment and available resources.

2.2.8 Risks

This concern deals with the risks of system development and operation. This concern clarifies which level of risk is acceptable. There may be various risks concerning the timely, cost effective deployment of a correctly functioning Core System.

2.2.9 Evolvability

This concern deals with the ability to extend and expand the system post-deployment to deal with new and potential unforeseen conditions or changes in its mission or scope.

2.2.10 Deployability

This concern deals with the issues surrounding deployment, including capital, human and other resource requirements.

2.2.11 Maintainability

This concern addresses issues of maintenance, in terms of resources required and impact on operations.

2.3 Stakeholder / Concern Matrix

This section summarizes the relations between stakeholders and concerns as a matrix, where rows correspond to concerns and columns to stakeholder roles. A shaded cell indicates that the concern is relevant to the stakeholder in that role.

Table 2-1: Stakeholder/Concern Matrix

Stakeholder / Concern	Mobile User	Field User	Center User	Operator	Acquirer	Maintainer	Developer	Manager	Tester	Policy Setter
Performance	🟪	🟪	🟪	🟪		🟪			🟪	
Interfaces	🟪	🟪	🟪	🟪		🟪	🟪		🟪	
Functionality	🟪	🟪	🟪	🟪		🟪	🟪		🟪	
Security	🟪	🟪	🟪	🟪		🟪	🟪	🟪		🟪
Organization/Resources				🟪	🟪	🟪		🟪		🟪
Appropriateness	🟪	🟪	🟪	🟪	🟪	🟪			🟪	🟪
Feasibility					🟪		🟪	🟪		
Risks	🟪	🟪	🟪		🟪		🟪	🟪		
Evolvability					🟪		🟪	🟪		
Deployability					🟪			🟪		🟪
Maintainability					🟪	🟪	🟪	🟪		🟪

3.0 ARCHITECTURAL VIEWPOINTS

The architecture of the Core System is described from multiple Viewpoints, each focusing on different concerns associated with the system.

A Viewpoint Specification is a specification of the rules and structure required to focus on particular concerns within a system. Each Viewpoint Specification includes a template from which to develop individual Views. A View is a representation of the system from the perspective of a related set of concerns.

Each Viewpoint Specification is intended to describe the system from a different perspective than the other viewpoints, but some specific areas of overlap exist to allow the elements in different Viewpoints to be related. Each Viewpoint exposes a different set of design concerns and issues, and each provides the means for reasoning about that aspect of the system.

Each Viewpoint Specification describes the Core System as a set of Objects and interactions among them. An Object is an abstract model of an entity in the system. Objects have behavior and state and are distinct from any other object. An object is characterized by whatever distinguishes it from other objects and by encapsulation, abstraction, and behavior. An example of an Enterprise object is an abstract model of an organization. Some attributes of this object include resources, policies etc. Objects defined in their primary Viewpoint often have corresponding objects in other Viewpoints. A Viewpoint Specification defines the rules for constructing Views of the system.

Encapsulation is the property that the information contained in an object is accessible only through interactions at the interfaces supported by the object. Because objects are encapsulated, there are no hidden side effects of interactions. That is, an interaction with one object cannot affect the state of another object without some secondary interaction taking place with that object. Thus, any change in the state of an object can only occur as a result of an external request of an object, an internal action of the object, or an interaction of the object with its environment.
Abstraction is the selective examination of certain aspects of a problem. The goal of abstraction is to isolate those aspects that are important for some purpose and suppress those aspects that are unimportant.
The **behavior** of an abstract data object is fully defined by a set of abstract operations defined on the object; the user of an object does not need to understand how these operations are implemented or how the object is represented.

A View is a representation of a specific system from the perspective of a set of concerns. Architecture views are representations of the overall architecture that are meaningful to one or more stakeholders in the system. They enable the architecture to be communicated to and understood by the stakeholders, so

Consider the analogy of a house to the Core System. A house has structural, electrical, plumbing, data and mechanical architectures. Each Viewpoint looks at exactly one of those architectures. Imagine that one could see one type of system by wearing an appropriate pair of glasses; e.g. electrical glasses, mechanical glasses etc. Looking in a window of the house with electrical glasses shows one View of the Electrical Viewpoint. In order to understand the entire house's electrical systems, one must look in every window with the electrical glasses. In order to understand the entire house, one must look in every window, and with every set of glasses.

Similarly, in order to understand the entirety of the Core System Architecture, one must understand every View of the Enterprise, Functional, Connectivity, Communications and Information Viewpoints.

they can verify that the system will address their concerns. Views are themselves modular and well-formed; each View is intended to correspond to exactly one Viewpoint and is constructed using the rules defined by that Viewpoint Specification.

The conceptual model of architectural description is captured in the figure below. In the figure, boxes represent classes of things. Lines connecting boxes represent associations between things. An association has two roles (one in each direction). A role can optionally be named with a label. The role from A to B is closest to B, and vice versa. A role can have a multiplicity, e.g., a role marked with "1.*" is used to denote *many*, as in a one-to-many or many-to-many association. A diamond (at the end of an association line) denotes a *part-of* relationship. The relationships illustrated above can be read as:

- A System has an Architecture
- A System has one or more Stakeholders
- An Architecture is described by one Architectural Description
- A Stakeholder has one or more Concerns.
- An Architectural Description identifies one or more Stakeholders
- An Architectural Description selects one or more Viewpoints
- An Architectural Description is organized by one or more Views
- A Concern is important to one or more Stakeholders
- A Viewpoint is addressed to one or more Stakeholders
- A Viewpoint is used to cover one or more Concerns
- A View conforms to a Viewpoint
- Views are part of an Architectural Description.

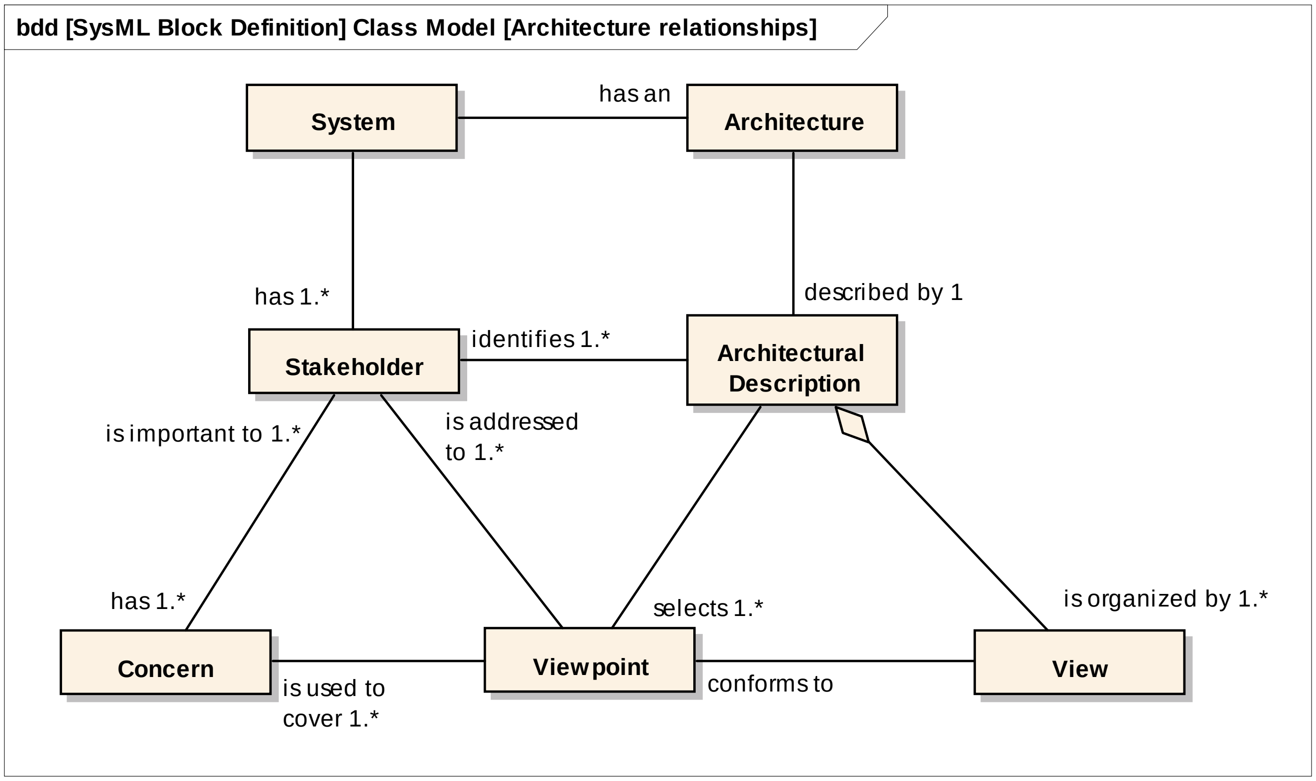


Figure 3-1: Conceptual Model of Architectural Description

Another way to picture the relationship between the Architecture, Viewpoints and Views is shown below. This figure identifies the five Viewpoints used to describe the Core System. There are multiple Views per Viewpoint. Only by considering all Views of each Viewpoint can one gain a complete picture the architecture.

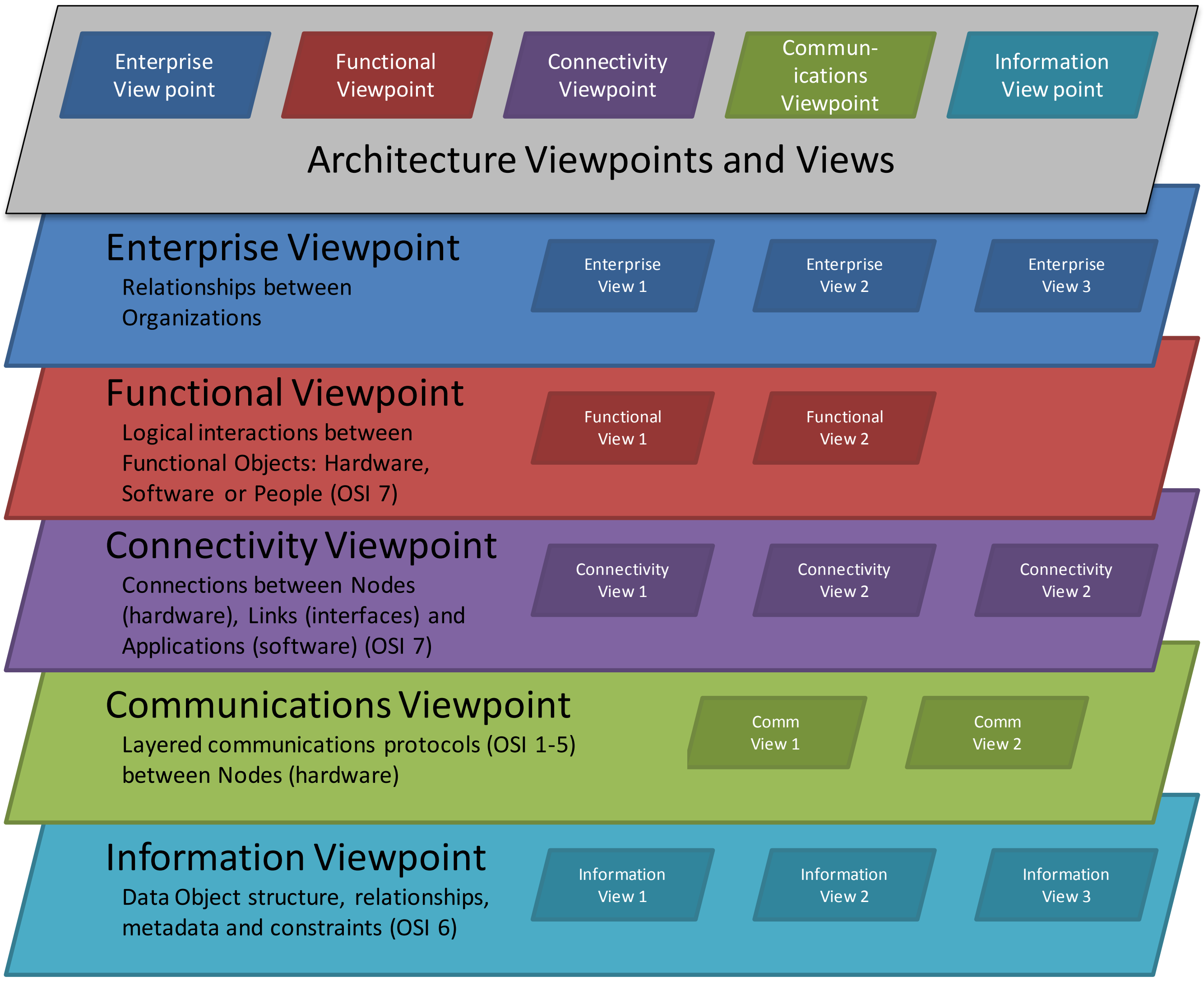


Figure 3-2: Core System Architecture Viewpoints

The Open Systems Interconnection (OSI) Model of communications layers has parallels to some of these viewpoints. While layers are not the same as viewpoints, viewpoints do tend to concentrate on particular layers. The Communications Viewpoint includes the OSI model layers 1-5 (Physical, Data Link, Network, Transport and Session), the Information Viewpoint covers layer 6 (Presentation) and the Functional and Connectivity Viewpoints align with Layer 7 (Application).

The architecture of the Core System is described by five Viewpoint Specifications, which are explained in the following subsections. Each viewpoint is documented as follows, where the letter *i* stands for the number of the viewpoint (1, 2, etc.).

Table 3-1: Viewpoint Description Template

Section #	Section Name	Description
3.i	**Name of Viewpoint**	Name of the Viewpoint
3.i.1	**Overview**	A brief overview of the viewpoint is provided
3.i.2	**Definition of Terms**	Defines terms used by the Viewpoint.
3.i.3	**Stakeholders Addressed**	Lists the stakeholders that this viewpoint is intended to address.
3.i.4	**Concerns Addressed**	Describes the concerns that this viewpoint is intended to address. Listed are questions that can be answered by consulting views that conform to this viewpoint.
3.i.5	**Objects and Relationships**	Lists the elements that may appear in the Viewpoint. Defines the objects, the attributes that may be attached to the object and describes rules for object interaction.
3.i.6	**Method(s) to Model/Represent Conforming Views**	Lists the methods and techniques that will be used to model or represent views conforming to this viewpoint. Includes legend for objects that may appear in Views.
3.i.7	**Viewpoint Source**	Provides a citation for the source of this viewpoint definition, if any.
3.i.8	**Security Issues**	Describes the security issues that this viewpoint is intended to address.
3.i.9	**Views Modeled**	Describes the Views that are modeled using the Viewpoint specification.

3.1 Global Definitions

These terms apply to more than one viewpoint:

A **policy** is the set of guidelines and constraints on the behaviors and states exhibited by the objects in the system.

A **role** describes the way in which an entity participates in a relationship; an object's set of behaviors and actions associated with the relationship of that object with other objects.

A **standard** is a formal specification that defines and governs functions and protocols at interfaces of a data system. It describes in detail the capabilities and establishes the requirements to be met by interfacing systems to achieve compatibility.

3.2 Enterprise Viewpoint

3.2.1 Overview

The **Enterprise Viewpoint** addresses the relationships between organizations and the roles those organizations play that involve various resources, including digital certificates, roadside equipment, Core System infrastructure, and communications network equipment. It also addresses personnel (including operators, users and support staff) that may be distributed among multiple organizations.

In the Enterprise Viewpoint, the Core System is depicted as a set of Enterprise Objects that interact with Enterprise Objects outside the Core. It focuses on the relationships between Core and external Enterprise Objects, but may include interactions between external Enterprise Objects if an alternative view shows that relationship impacting the Core. Enterprise Objects representing system elements that have significant resources may appear in an Enterprise View as Facilities.

The relationships between Enterprise Objects are largely determined by the Enterprise's roles, responsibilities, policies and goals, and not by Core System policies or goals. The relationships between Enterprise Objects will therefore depend on the responsibilities of the Enterprises involved and the roles they choose to play in implementing the Core.

3.2.2 Definition of Terms

An **Enterprise Object** represents an entity that is governed by a single authority that has its own objectives and policies for operating the object. An Enterprise Object may be a component of another larger Enterprise Object, which may in turn be a component of a third, even larger, Enterprise Object. Enterprise Objects may participate wholly or in part in other Enterprise Objects.

A **Facility** is a physical infrastructure element that supports the use of services and other resources.

A resource in general terms is anything available to a system that can support the achievement of objectives; any physical or virtual element that may be of limited availability within a system. In this context a **Resource** is an Enterprise Object that has some role, offers services, and performs some action within a system. A resource may serve more than one activity.

A **domain** is an Enterprise Object that is under single organizational, administrative or technical control.

A **federation** is a group of domains that come together to share resources while each domain retains its authority over its own resources. Federations are governed by negotiated agreements.

An organizational Enterprise Object may own a facility or resource Enterprise Object. **Ownership** means having administrative and fiscal responsibility for the owned element and the right to exclusively control and use that which is owned for one's own purposes. It is the state or fact of having exclusive possession or control of some object, facility, intellectual property or some other kind of property. Not every organizational object owns facilities or resources. Some resources are owned by one organization and used by others. The term **cross support** is used to describe an agreement between two or more organizations to exploit the technical capability of interoperability for mutual advantage, such as one organization offering support services to another in order to enhance or enable some aspect of a mission.

3.2.3 Stakeholders Addressed

Mobile User, Field User, Center User, Acquirer, Operator, Maintainer, Developer, Manager and Policy-Setter.

3.2.4 Concerns

Security	What entities are involved in the distribution of digital certificates, and what roles do those entities have? What entities are involved in the detection of misbehavior by System Users, and what roles do those entities have?
Organization/Resources	Who needs to contribute resources to Core System development? What resources are required to support Core System development? What Core System resources are required to support external application development?
Risks	What relationships between Core Enterprise Objects and external Enterprise Objects provide risks to Core System development, deployment and/or operations? What steps can be taken to lessen risks that are a function of Enterprise relationships?
Evolvability	How do the relationships between Core Enterprise Objects and external Enterprise Objects need to change to support the integration of new Enterprises? Specifically, what is the decision mechanism for integrating new Enterprises and modifying roles of existing Enterprises?
Deployability	Who needs to contribute resources to Core System deployment? What resources are required to support Core System deployment? What resources will be required to develop the Core System's functionality?
Maintainability	Who needs to contribute resources to Core System maintenance activities? What resources are required to support Core System maintenance?

3.2.5 Objects and Relationships

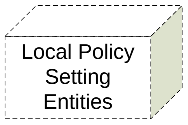
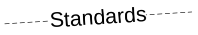
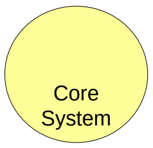
The following elements may appear in the Enterprise Viewpoint:

- Enterprise Objects:
    - Types: Organizations and Resources
    - Attributes: Name, type, role, objectives, location (relative or absolute), members, resources, interaction modes, requirements and constraints
    - Inputs: Requirements, agreements, contracts, funding, policies, rules, purpose
    - Outputs: Requirements, policies, rules, agreements
- Domains: boundaries of responsibility or ownership
- Relationships: ownership, membership, participation, roles, contractual
- Information: instances of documents, agreements, contracts, policies, requirements, objectives, goals, interface specifications. Formal definition of data to be exchanged may be found in the Information Viewpoint.

3.2.6 Method(s) to Model/Represent Conforming Views

Enterprise View diagrams use the graphical objects defined in Table 3-2.

Table 3-2: Enterprise View Graphical Object Definitions

	Enterprise Objects and Facilities are represented by 3-dimensional shaded boxes drawn with dashed lines, with the name of the Enterprise or Facility on the front face. Objects are named as descriptively as possible with regard to the role they play in the View.
	A logical relationship between Enterprise Objects indicated the exchange of information. The example here indicates that standards documents are being exchanged.
	Domains are modeled with colored circles.

3.2.7 Viewpoint Source

This Viewpoint is based on the Enterprise Viewpoint documented in CCSDS 311.0-M-1.

3.2.8 Security Issues

In the Enterprise Viewpoint the security issues which must be addressed include organizational roles, policies, trust relationships, domain boundaries and cross-support security agreements. The implementation mechanisms to enforce these rules and agreements are detailed in other viewpoints.

Organizational roles define the behaviors and actions that an entity may undertake in association with the relationships it has with other entities. This implies some limits on the entity's behavior; if behaviors are not followed as the architecture describes, this may result in a security or functional problem.

Similarly, policies established by one entity may constrain or place requirements on the roles of other entities. Enforcement of adherence to these policies and thus fulfilling roles requires system level decisions. Is policy enforcement going to be undertaken by an over-arching entity, or will entities simply dissolve relationships if the relevant partner does not fulfill his role? In either case, if the role in question is one that is critical to system operation, who will step in to fill the void? These are questions that cannot be answered by the architecture, but they can be illustrated.

Trust relationships are at the heart of the Core System; the Core's main job is facilitating trust between System Users. Trust between entities that implement, operate and develop Cores is a different thing entirely. The trust that the Core ensures is between devices; hardware and software devices for which protocols can be defined that provide a high confidence in the trustworthiness of communicating partners. Trust between entities must be established by humans, established by agreement and possibly documented in a contractual relationship. Such trust relationships can be addressed and illustrated in the architecture, but must be implemented by leaders of the respective entities involved.

Domain boundaries and cross-support security agreements follow in the wake of trust relationships, roles and policies governing operational scope and security. A federation of Core Systems, where a Core is a singular domain, may be established by establishing trust between Cores, agreeing on and implementing policies that are consistent between Cores, including the conventions for defining and modifying Core boundaries. This federated environment will enable security agreements between Cores, and ensure a consistent approach to the handling of security issues such as certificate distribution and revocation across all Cores.

Details concerning these security issues are addressed individually in the text associated with each view.

3.2.9 Views Modeled

Four enterprise views are modeled:

1. Enterprise View – DSRC Security Credentials Configuration: addresses the distribution of digital certificates required to implement IEEE 1609.2.
2. Enterprise View – Development: addresses the development of Core System software as well as 3rd party application software as it relates to the Core.
3. Enterprise View – Maintenance: addresses the maintenance of the Core System.
4. Enterprise View – Governance: addresses management and leadership decision-making related to defining Core System use and management.

3.3 Functional Viewpoint

Functional Viewpoint

Logical interactions between Functional Objects: Hardware, Software or People (OSI 7)

3.3.1 Overview

The Functional Viewpoint separates the analysis of abstract functional elements and their logical interactions from the engineering concerns of how functions are implemented, where they are allocated, how they transfer information, which protocols are used, and what method is used to implement them. Keeping the analysis of functional behavior required of a system separate from the details of how to implement it provides a degree of freedom to do design trade-off studies separate from functional design.

The Functional Viewpoint of the Core System focuses on the behavior, structure, and interaction of the functions performed by the system. This Viewpoint addresses Functional Objects, their behavior, the logical connections between them, the information they exchange, and their logical interfaces and interactions.

The behavior of a Function is the set of actions performed by this element to achieve an objective. A **Functional Object** performs actions to achieve an objective of the Core System or to support actions of another Functional Object. This may involve data transformation, generation, or processing in performing those actions. Functional Views define Functional Objects to control and manage system behavior, such as monitoring, and other active control elements that are part of describing the functional behavior of the system. They also describe processing functions and the logical flows of information among these Objects.

3.3.2 Definition of Terms

A **Functional Object** is an abstract model of a functional entity that receives requests, performs actions, and generates or processes data. Functional objects that only transport data are called Protocol Entities and are treated explicitly in the Communications Viewpoint. A Functional Object may be a component of another larger Functional Object, which may in turn be a component of a third, even larger, Functional Object. A Functional Object may be implemented by people but most are implemented as software and/or hardware. Implementations of Functional Objects are basic Engineering Objects which are described in the Connectivity Viewpoint.

3.3.3 Stakeholders Addressed

Mobile User, Field User, Center User, Acquirer, Operator, Maintainer, Developer, Manager, and Tester.

3.3.4 Concerns

Performance	Can the Core System provide services with sufficient responsiveness to enable System User applications? Are there any Functional Objects in the Core System that do not have performance requirements in the System Requirements Specification (SyRS) and need them? Can the Core meet all of the performance requirements defined in the SyRS?
Interfaces	How difficult is it to develop applications that use Core System interfaces?

	How flexible are Core System data distribution interfaces? Does the Core System meet all of the interface requirements defined in the SyRS?
Functionality	How does the Core System monitor and enable control of the services it provides?
Security	What entities are involved in the distribution of digital certificates, and what roles do those entities have? What entities are involved in the detection of misbehavior by System Users, and what roles do those entities have?
Appropriateness	Does the Core System meet all of the needs defined in the ConOps? Does the Core System meet all of the functional requirements defined in the SyRS? Are there any Functional Objects in the Core System that do not have functional requirements in the SyRS?
Feasibility	Are Core System services feasible to develop given current technology and resources?
Evolvability	Will Core System services remain relevant in the light of foreseeable advances in technology and services offered by other systems? How easily can the Core's functionality be expanded to cover new needs if they arise?
Maintainability	Can the Core's functionality be sustained while maintaining its services and systems?

3.3.5 Objects and Relationships

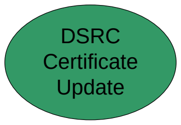
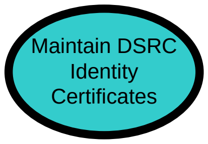
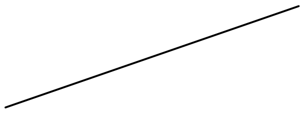
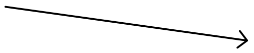
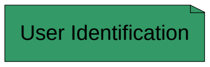
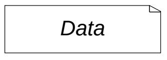
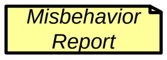
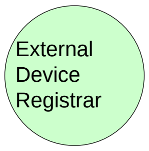
The following elements may appear in the Functional Viewpoint:

- Functional Objects:
    - Types: Data source, data sink, data transformer, control, planning, monitoring, analysis
    - Attributes: Name, role, behavior, interface definition, data types handled, interaction mode, allocated requirements and constraints
    - Inputs: Control request, service request, data
    - Outputs: Data, service request, control request
- Logical Links: connections between Functional Objects, associated with behavior and object properties
- Relationships: configuration, precedence, control flows, data flows, management flows
- Information: representations of data that are exchanged between Functional Objects. Formal definition of data to be exchanged may be found in the Information Viewpoint.

3.3.6 Method(s) to Model/Represent Conforming Views

Functional View diagrams use the graphical objects defined in Table 3-3.

Table 3-3: Functional View Graphical Object Definitions

	Functional Objects are represented by ovals, with the name of the object in the oval. Objects are named as descriptively as possible with regard to the role they play in the View.
	Functional Objects that store encrypted data are depicted with bold outlines.
	A logical relationship between Functional Objects. Relationships may be tagged with content type (control, service, data). Relationships may be associated with an Information Object.
	A directional external data flow between a Functional Object and an external Object. Directional external flows may be unidirectional or bidirectional.
	An Information Object that is exchanged as part of a relationship between Functional Objects. A brief description of the Information Object's contents, source and destination may appear in the functional view. Formal definition of the Information Object may be found in the appropriate Information View.
	An Information Object that is digitally signed is named with *italics text.*
	An Information Object that is encrypted is drawn with bold outlines. A signed and encrypted object has italics text with bold outlines.
	Domains are modeled with colored circles.

3.3.7 Viewpoint Source

This Viewpoint is based on the Functional Viewpoint documented in CCSDS 311.0-M-1.

3.3.8 Security Issues

The Functional Objects and services that are used to implement security policies and approaches are defined in the Functional Viewpoint; these include access control interfaces on functions and specific functional elements such as authentication, encryption, certificate distribution, certificate revocation and key management. Some of these may be shown as Functional Objects in their own right (e.g., Public Key Infrastructure [PKI] management function), or just as attributes of other Functional Objects (e.g., access control on a management or control function).

Details concerning these security issues are addressed individually in the text associated with each view.

3.3.9 Views Modeled

Seven functional views are modeled:

1. Functional View – Top Level: provides a high level functional view considering the subsystems defined in the Concept of Operations.
2. Functional View – Data Distribution: addresses the Core System's implementation of publish and subscribe.
3. Functional View – System Monitor and Control: addresses the Core System housekeeping-type operations.
4. Functional View – DSRC Certificate Distribution: addresses the distribution of digital certificates necessary to implement 5.9 GHz DSRC and IEEE 1609.2.
5. Functional View – X.509 Certificate Management: addresses the distribution of digital certificates for non-DSRC System Users.
6. Functional View – Core Decryption: addresses decryption of data sent by a System User in encrypted form and intended for Core System use.
7. Functional View – Networking: addresses the Core's networking functionality.

One additional functional view will be developed in a future version of this document:

Functional View – Core Backup

3.4 Connectivity Viewpoint

Connectivity Viewpoint

Connections between Nodes (hardware), Links (interfaces) and Applications (software) (OSI 7)

3.4.1 Overview

The Connectivity Viewpoint represents physical elements that operate in the field and the back office, where connections between elements, the physics of motion, and interactions with forces in the external environment are considered. The Connectivity Viewpoint deals with the composition of these physical elements and their connections and interactions. For analysis of the Core System and its relationship with other elements of the connected vehicle environment, the focus is on nodes, links, computational and data transport functions.

Mobile Users are in motion and consequently connectivity issues exist associated with the wireless communications media used to interact with the Core System or other users.

The Connectivity Viewpoint also includes all of the other aspects of Core System dealing with the composition of physical elements, their physical connections, and the allocation of functionality to those elements. The physical elements include application servers, data stores, wired and wireless links.

The **Connectivity Viewpoint** is an engineering view that shows Engineering Objects, which may be hardware or software. The Connectivity Viewpoint is focused on a **Node** and **Link** view of a system, the composition of the Nodes, the physical connections among Nodes, their physical and environmental constraints, and their physical dynamics. The Connectivity Viewpoint also describes how the abstract functional design described in the Functional Viewpoint is to be implemented as software Engineering Objects, i.e., applications or software components or hardware Engineering Objects, and how these are allocated on the major hardware Engineering Objects (Nodes) of the system.

3.4.2 Definition of Terms

An **Engineering Object** is an implementation or realization of some abstract function. It may be implemented as hardware (Node) or as software (application or software component).

A **Node** is a physical hardware Engineering Object that is a run-time computational resource and generally has at least memory and processing capability. Run-time software Engineering Objects reside on nodes. A Node has some well-understood, possibly rapidly moving, location. A Node may be composed of two or more (sub) Nodes.

A **Link** is the locus of relations among Nodes. It provides interconnections between Nodes for communication and coordination. It may be implemented by a wired connection or with some radio frequency (RF) or optical communications media. Links implement the primary function of transporting data. Links connect to Nodes at a Port.

A **Port** is the physical element of a Node where a Link is connected. Nodes may have one or more Ports. Each Port may connect to one or more physical Ports on (sub) Nodes that are contained within the Node.

An **Application** consists of one or more pieces of software designed to perform some specific function; it is a configuration of interacting Engineering Objects.

3.4.3 Stakeholders Addressed

Acquirer, Center User, Mobile User, Field User, Maintainer, Developer, and Tester.

3.4.4 Concerns

Performance	Can the Core meet all of the performance requirements defined in the SyRS? Are there any interfaces or nodes that do not have performance requirements defined, and need to have them?
Interfaces	Can the Core System meet all of the interface requirements defined in the SyRS?
Security	What physical elements are involved in the distribution of digital certificates, and what are their roles? What physical elements are involved in the distribution of certificate revocation lists and what are their roles? How are the Core System components physically secured?
Feasibility	Are Core System services feasible to develop given current technology and resources?
Evolvability	Is the structure of the Core System sufficiently flexible and scalable to enable changes to cover new needs if they arise?
Deployability	Are the Engineering Objects that make up the Core System practical to deploy in the projected deployment environment?

3.4.5 Objects and Relationships

The following elements may appear in the Connectivity Viewpoint:

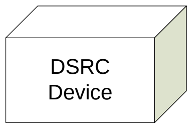
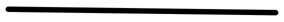
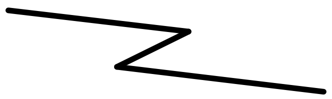
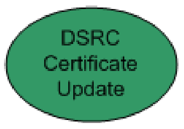
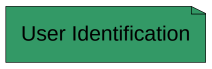
- Engineering Objects:
    - Nodes (hardware Engineering Objects)
        - Types: hardware objects, ports
        - Attributes: Name, type, location, laws of motion, available resources, physical interfaces, capabilities (e.g., processing, memory, bandwidth, throughput, etc.), allocated functions and constraints
        - Inputs/Outputs: Links to/from other Nodes
    - Links (connections between Nodes)
        - Types: Wireless, physical link
        - Attributes: Name, type, end-points (ports), physical interfaces, performance, access, ownership
    - Applications (software Engineering Objects)
        - Types: software engineering objects

    - Attributes: name, type, algorithms, implemented functions, allocation to nodes, required resources, implemented interfaces, implementation constraints (e.g., language, operating system, framework, etc.)
- Relationships: composition, interfaces, constraints, configuration
- Environment: physical environments, physical interactions and effects
- Information: defined representation of data that are exchanged among Engineering Objects. Formal definition of data to be exchanged may be found in the Information Viewpoint.

3.4.6 Method(s) to Model/Represent Conforming Views

Connectivity View diagrams use the graphical objects defined in Table 3-4.

Table 3-4: Connectivity View Graphical Object Definitions

	Nodes are represented by 3-dimensional shaded boxes drawn with solid lines, with the name of the Node on the front face. Nodes are named as descriptively as possible with regard to the role they play in the View.
	A link between Nodes. The link may be tagged with content type (control, service, or data). This link is one that could be implemented using a physical, wired connection.
	A link between Nodes. The link may be tagged with content type (control, service, or data). This link is one that must be implemented using a wireless connection.
	Applications external to the Core System and Core Functional Objects are represented by ovals with the name of the object in the oval. The Core Functional Object definition may be found in the appropriate Functional View.
	An Information Object that is exchanged as part of a relationship between Engineering Objects. The Information Object definition may be found in the appropriate Information View.
	Ports are modeled as small boxes.

3.4.7 Viewpoint Source

This Viewpoint is based on the Connectivity Viewpoint documented in CCSDS 311.0-M-1.

3.4.8 Security Issues

In the Connectivity Viewpoint security issues are dealt with by the physical elements that are used to implement security policies and barriers. These include routers and firewalls, hardware encryption devices, and physical boundaries such as shielded rooms. The protocol entities that may implement elements of security functionality such as security protocols will be addressed in the Communications Viewpoint.

3.4.9 Views Modeled

Two connectivity views are modeled:

1. Connectivity View – High Level: addresses the basic component structure of the Core System.
2. Connectivity View – Core System Function Allocation: addresses functional allocation to Core components.

3.5 Communications Viewpoint

Communications Viewpoint
Layered communications protocols (OSI 1-5) between Nodes (hardware)

3.5.1 Overview

The Communications Viewpoint defines the layered sets of communications protocols that are required to support communications among the Engineering Objects that compose the Core System and associated users. These protocols need to meet the requirements on performance and the constraints imposed by physical connectivity, environmental and operational challenges, and relevant policies (such as the provision of anonymity for mandatory data provision).

The Communications Viewpoint is a system engineering and technical view that focuses on the mechanisms and functions required to design and implement protocols and communications standards, including implementation choices, and specification and allocation of communications functionality to engineered components of the system.

This Viewpoint is orthogonal to the first three viewpoints. Traditionally, it provides details on layers one through five of the OSI seven-layer model.[3]

In the Communications Viewpoint the Core System is depicted with Communications Objects that are called Protocol Entities. For context, these are often shown along with representations of the Nodes, Links, and software Engineering Objects that are defined more fully in the Connectivity Viewpoint. The Communications Viewpoint describes the protocols that are required for the software Engineering Objects actually to communicate with one another and supports descriptions of the end-to-end information system.

3.5.2 Definition of Terms

A **Protocol Entity** performs actions to exchange or transfer data (as distinguished from a Functional Object that generates or processes data). Protocol Entities are used to support interactions between two Engineering Objects or among groups of Engineering Objects that are contained in separate Nodes. Protocol Entities are often shown as two peer entities communicating with each other over a Link between connected Nodes. The Engineering Objects that use protocol services may be implemented in hardware or software, and the Protocol Entities themselves may be implemented in hardware or software.

3.5.3 Stakeholders Addressed

Mobile User, Center User, Field User, Operator, Acquirer, Maintainer, Developer, Tester, and Policy Setter.

3.5.4 Concerns

Performance	Do the communications protocols allow the Core to meet the performance requirements defined in the SyRS?
Interfaces	Do communications protocols allow the Core to meet the interface requirements defined in the SyRS?

[3] The Functional and Connectivity viewpoints are directly related to the Application layer, and the following Information Viewpoint is most closely related to the Presentation layer.

Functionality	What functionality exists in the communications protocols used by the Core? What reliability features are included in the communication protocols?
Security	What provisions for ensuring the security of communications by System Users are included in the communications protocols?
Organization/Resources	What resources are required to develop the communications protocols that the Core System needs?
Feasibility	Are the communications protocols required by the Core System feasible to specify, develop, and deploy?
Risks	If communications protocols chosen are developed by independent bodies are there any risks associated with changes to those standards that may impact the applicability of those standards?
Deployability	Are the communications protocols required by the Core practical to deploy from a capital and human resource perspective?

3.5.5 Objects and Relationships

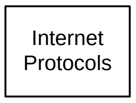
The following elements may appear in the Communications Viewpoint:

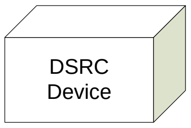
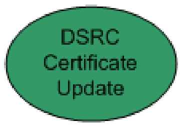
- Protocol Entities:
    - Types: protocol purpose (e.g., coding, link, network, transport, etc.)
    - Attributes: Name, type, capabilities (e.g., in order, once only, bandwidth efficient, error correcting, delay tolerant), constraints, services (offered, required), interface signature (requests, indications, responses, confirmations), standard reference identifier, standards organization, management interface
    - Inputs and Outputs defined above under attributes
- Protocol design specification elements: state machine, acknowledgement
- Nodes and Links: representations of the physical elements from the Connectivity Viewpoint, for context
- Software Engineering objects: representations of implemented functions from the Connectivity Viewpoint for context

3.5.6 Method(s) to Model/Represent Conforming Views

Communications View diagrams use the graphical objects defined in Table 3-5.

Table 3-5: Communications View Graphical Object Definitions

	Protocol entities are represented by rectangles with the name of the entity inside.

	Nodes are represented by 3-dimensional shaded boxes drawn with solid lines, with the name of the Node on the front face. The Node definition may be found in the appropriate Connectivity View.
———	The Link between nodes that connects to the lowest layer protocol entity. Extensions of this link go between protocol entities, and between protocol entities and Software Engineering Objects.
- - - - - -	The logical Link between Protocol Entities operating at the same communications layer, or between Software Engineering Objects.
	Software Engineering Objects are represented by ovals, with the name of the object in the oval. Software Objects in the Communications View also appear as Functional Objects in a Functional View, and may be in a Connectivity View. The Functional Object definition may be found in the appropriate Functional View.

State machines each have their own diagram, and are modeled as SysML state machine diagrams.

3.5.7 Viewpoint Source

This Viewpoint is based on the Communications Viewpoint documented in CCSDS 311.0-M-1.

3.5.8 Security Issues

Certain functions for implementing data system security may be allocated to the Communications Viewpoint. These will typically include network layer, transport layer and session layer security protocols. These functions need to be traced to the SyRS. Different physical communications media may require different upper-layer protocols.

Implementation of security protocols between Mobile Users and the Core System, and Mobile Users and other System Users, is a primary concern tracing back to many of the needs documented in the ConOps. Communications protocols will realize much of this functionality, which needs to be documented here.

3.5.9 Views Modeled

One communications view is modeled:

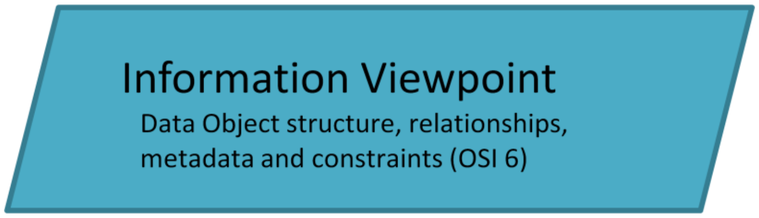
1. Communications View – Mobile DSRC Device and Core: addresses Mobile Users communicating with the Core System using DSRC communications.

Three additional communications views will appear in a future version of this document:

2. Communications View – Mobile Cellular User and Core: addresses Mobile Users communicating with the Core using cellular communications.
3. Communications View – Core and Core: addresses Cores communicating with one another.

3.6 Information Viewpoint

3.6.1 Overview


Information Viewpoint
Data Object structure, relationships, metadata and constraints (OSI 6)

The Information Viewpoint describes the data objects that are passed among the elements in the Core System. These data objects may have different elements, structures, semantics, relationships, and policies. The Information Viewpoint is used to address the data architecture and definition aspects of the system. Representations of the Information Objects that are fully defined in this Viewpoint appear in other Viewpoints. They are managed (i.e., stored, located, accessed and distributed) by information infrastructure elements and also shown as being passed among enterprise, functional and application entities.

The Information Viewpoint specification of the Core System focuses on the information used by the system. This is limited to views of information content and the relationships among information elements.

3.6.2 Definition of Terms

Information Objects are descriptions of data along with the necessary structure and syntax to allow interpretation and use of these Objects. An Information Object may also have associated metadata, and information views may define the relationships among Data Objects, rules for their use and transformation, and policies on access.

Metadata is ‘data about data’, the information that describes content. It is information about the meaning of data, as well as the relationships among Data Objects, rules for their use and transformation, and policies on access.

An **Information Package** consists of a primary Information Object, with associated Metadata necessary to use the Information Object. The Information Package has associated Packaging Information used to delimit and identify the primary Information Object and Metadata.

3.6.3 Stakeholders Addressed

Mobile User, Field User, Center User, Operator, Acquirer, Developer, and Tester.

3.6.4 Concerns

Security	Do any Information Objects potentially impact the privacy of System Users?
Appropriateness	Do the Information Objects convey information sufficient to realize Core System functionality as defined in the SyRS?

3.6.5 Objects and Relationships

The following elements may appear in the Information Viewpoint:

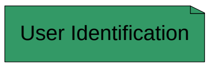
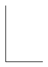
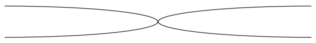
- Information Objects:
  - Types: data, metadata, schema, model
  - Attributes: Name, type, length, semantics, rules, policies
  - Inputs and Outputs defined above under attributes

- Relationships: aggregates, transformation
- Constraints: permanence, policies

3.6.6 Method(s) to Model/Represent Conforming Views

Information View diagrams use the following graphical objects:

Table 3-6: Information View Graphical Object Definitions

	An Information Object.
	Aggregation relationship. Parent Information Object will be at the top, child object to the right and below.
	Transformation relationship. Initial Information Object will be on the left, resultant transformed object on the right.

3.6.7 Viewpoint Source

This Viewpoint is based on the Information Viewpoint documented in CCSDS 311.0-M-1.

3.6.8 Security Issues

Information Objects may contain information that needs to be protected from authorized access. These objects must be identified so that appropriate security mechanisms can be architected in the Communications and Functional Viewpoints.

3.6.9 Views Modeled

One Information View is modeled:

1. Information View – Top Level External Objects: addresses the Information Objects that the Core sends to and receives from System Users.

An additional Information View will be modeled in a future version of this document:

Information View – Top Level Internal Objects, which will address the Information Objects shared between Core Functional Objects.

4.0 ARCHITECTURAL VIEWS

This section of the System Architecture Document contains views for each viewpoint listed in Section 3. Each view is documented as follows:

Section #	Section Name	Description
4.x	**Name of Viewpoint**	Name of the Viewpoint x
4.x.y	**Name of View**	Name of the View y of Viewpoint x
4.x.y.1	**Introduction**	A brief introduction to the view is provided
4.x.y.2	**Object Definitions and Roles**	Defines all of the objects that appear in the view, their function and or purpose, and relationship to other objects.
4.x.y.3	**View Description**	A graphical representation of the system is constructed with the methods of the associated viewpoint, accompanied by a textual discussion.
4.x.y.4	**Configuration Information**	Identifies which other views need to be considered when managing updates to this view. Changes to this view may necessitate changes to these other views.
4.x.y.5	**Alternatives Considered**	Describes alternatives to the view. In some cases this includes a new diagram and object definitions. Also includes a discussion of advantages or disadvantages to the alternative.

This June 13 draft of the System Architecture is a work-in-progress document. The viewpoints documented here vary in their levels of completeness. Enterprise, Functional and Connectivity views are the most complete. Much of what will eventually appear in the Information Viewpoint can be gleaned from the associated information object statements in the Functional views.

The views documented here define the architecture for the Core System to the degree necessary to evaluate alternatives.

The following views include alternatives:

Enterprise Viewpoint: Enterprise View – DSRC Security Credentials Configuration
Enterprise Viewpoint: Enterprise View – Governance
Functional Viewpoint: Functional View – Data Distribution
Functional Viewpoint: Functional View – DSRC Certificate Distribution
Functional Viewpoint: Functional View – X.509 Certificate Management
Functional Viewpoint: Functional View – Core Decryption
Functional Viewpoint: Functional View – Networking

Connectivity Viewpoint: Connectivity View – High Level
Connectivity Viewpoint: Connectivity View – Core System Function Allocation
Communications Viewpoint: Communications View – Mobile DSRC Device and Core

In every case, the recommended view is the one documented in sections 4.x.y.2-3. In some cases, alternatives are explicitly stated to be compatible with the recommended view; in these cases, it is recommended that the alternative view also be adopted (i.e., two recommended views).

In some cases the selection of an alternative view from one viewpoint forces the selection of an alternative view from another viewpoint. This is explicitly stated in the affected viewpoints.

ALTERNATIVES ANALYSIS

The following questions are intended to help in evaluating alternatives, particularly if a view is modified or if an alternative view is selected, there will be impacts to other aspects of the *connected vehicle* environment. For some of these views changes may result in substantially modified operation of the Core System and the *connected vehicle* environment as a whole may be substantially affected. Considering these questions should help choose the architectural views for the Core System:

1) Should the Core function as a DSRC Certificate Authority (CA), Registration Authority (RA)? Should external entities operate as CA or RA? The recommendation is that one Core operates as CA, all other Cores operate as RA, and that external entities be allowed to operate as RA. The relevant architecture views are:
   Enterprise Viewpoint: Enterprise View – DSRC Security Credentials Configuration
   Functional Viewpoint: Functional View – DSRC Certificate Distribution

2) What makes a system a Core System? Who says who can be a Core? The recommendation is that a Core Certifying Body be created to validate Cores against this SAD and the SyRS. The relevant architecture views are:
   Enterprise Viewpoint: Enterprise View – Governance
   Enterprise Viewpoint: Enterprise View – DSRC Security Credentials Configuration
   Functional Viewpoint: Functional View – DSRC Certificate Distribution
   Functional Viewpoint: Functional View – X.509 Certificate Management

3) Should the Core System parse data intended for publish/subscribe and provide only the data elements that data consumers need, or forward them the entire block of data that includes both data they want and data they do not need? The recommendation is that data be parsed and only desired data provided to data consumers. The relevant architecture view is:
   Functional Viewpoint: Functional View – Data Distribution

4) Should private networks connect directly to the Core? The recommendation is that private networks be allowed to connect directly to the Core. The relevant architecture view is:
   Connectivity Viewpoint: Connectivity View – High Level

Each of the relevant architecture views above contains a description of the recommended approach along with one more alternatives that were analyzed during the development.

4.1 Enterprise Viewpoint

The Enterprise Views includes two views with alternatives, one of which, DSRC Security Credentials Configuration, has an impact on the Functional Viewpoint. Enterprise Views illustrated in the SAD were selected to illustrate the number and variety of entities that could be involved in operations, deployment and management of Core Systems.

4.1.1 Enterprise View – DSRC Security Credentials Configuration

4.1.1.1 Introduction

The safety-critical nature of many DSRC/WAVE applications makes it vital to protect messages from attacks such as spoofing, alteration, and replay. To accomplish this, the Core System distributes security credentials to DSRC devices.

Standards bodies like Internet Engineering Task Force (IETF) and IEEE have drafted standards for configuration and distribution of security credentials. Standards Development Organizations (SDOs) have been developing security standards suitable for implementing certificate-based trust schemes using DSRC-based communications. Enterprises like VeriSign, Entrust, and Microsoft have developed security products based on IETF standards. Implementers of the Core System will need to come up with policies related to security credentials for devices operating in their domain. Core deployers can procure Commercial-off-the-Shelf (COTS) products for managing security credentials or farm out this activity to enterprises like VeriSign or Entrust. This view depicts the Enterprise Objects involved in the mission of configuring Security Credentials for a DSRC Device.

4.1.1.2 Object Definitions and Roles

Certificate Authority (CA): The CA is the External Support Entity that receives requests and distributes certificates to Device Manufacturers. The CA receives DM authorization information from the Core System Manager. The CA receives policies, practices and standards for certificate implementation from the Core System Deployer.

Core System Deployer: The Core System Deployer provides the policies, best practices and standards for certificate format and procedures to the CA. It receives standards from Standards Bodies, and policies and practices from the entities that operate the roadway infrastructure in the area over which its Core operates.

Core System Manager: The Core System Manager certifies that a given Device Registrar is authorized to distribute certificates, and then passes that authorization on to the CA.

DSRC Device: The Device that wants to participate in the connected vehicle environment. The Device receives certificates from the Device Registrar.

DSRC Device Registrar: The manufacturer of the on-board device. The manufacturer enters into an agreement with the Device Owner so the Device Registrar can obtain and configure certificates on the device. The Device Registrar is authorized by the Core System Manager, to whom the Device Registrar must prove he is trustworthy and qualified to configure certificates. The Device Registrar requests and receives certificates from the Certificate Authority.

DSRC Device Owner: The owner of a device that wants to participate in the connected vehicle environment. The owner enters into an agreement with the manufacturer of the connected vehicle device for that manufacturer to act on his behalf and obtain certificates.

Field Node Owner/Operator: The owner and operator of the field infrastructure used by the DSRC Device Owner to access the DSRC Device Registrar. This could be a cellular provider or DSRC field infrastructure manager.

Local Policy Setting Entities: This body provides policies and best practice recommendations for Core System deployment and operations as well as policies for local user authorizations (i.e., who can do what). Entities filling this role could include local DoTs, local, county and state government, regional transportation authorities and non-governmental entities such as professional trade associations.

Standards Bodies: Standards bodies develop standards and make those available to the Core System Deployer.

4.1.1.3 View Description

Distribution of security credentials is a task of the Core System, but in this view much of the activity is delegated to external support entities: the Certificate Authority and Device Manufacturer. The Core System Manager authorizes the Device Registrar as qualified to distribute certificates on behalf of the Core. To ensure compatibility between Cores this authorization will have to be uniform; all Cores will have to agree to and recognize that the Device Registrar is authorized to distribute certificates.

The Device Registrar will enter into an agreement with the owner of the device to provide and install certificates. The Device Registrar can then request certificates from the CA. The CA will have to recognize that the Device Registrar is authorized to receive such certificates; such authorization must come from the Core System Manager. If the CA agrees that the request is valid and the Device Registrar is authorized to request certificates, it will distribute certificates.

If all Core Systems have the same standards for recognizing Device Registrars, then the Device Registrar need only make certificate requests of one Core. Otherwise the Device Registrar would have to communicate with every Core whenever it wished to acquire certificates for a DSRC device. Since there are likely to be multiple Device Registrars and multiple Cores, this will lead to a web of relationships between entities that must be continuously exercised. This operational complexity can be avoided through standardization and certification of Cores, which is covered under Enterprise View – Governance.

Other entities involved provide the basis for exchanging data. Policies and best practice procedures come from the trade organizations and bodies (often but not always governmental) responsible for the geographic area over which the Core System operates. Standards bodies provide the standards that define the interfaces between the CA and Core System Manager, Core System Manager and Device, and also the format and characteristics of certificates.

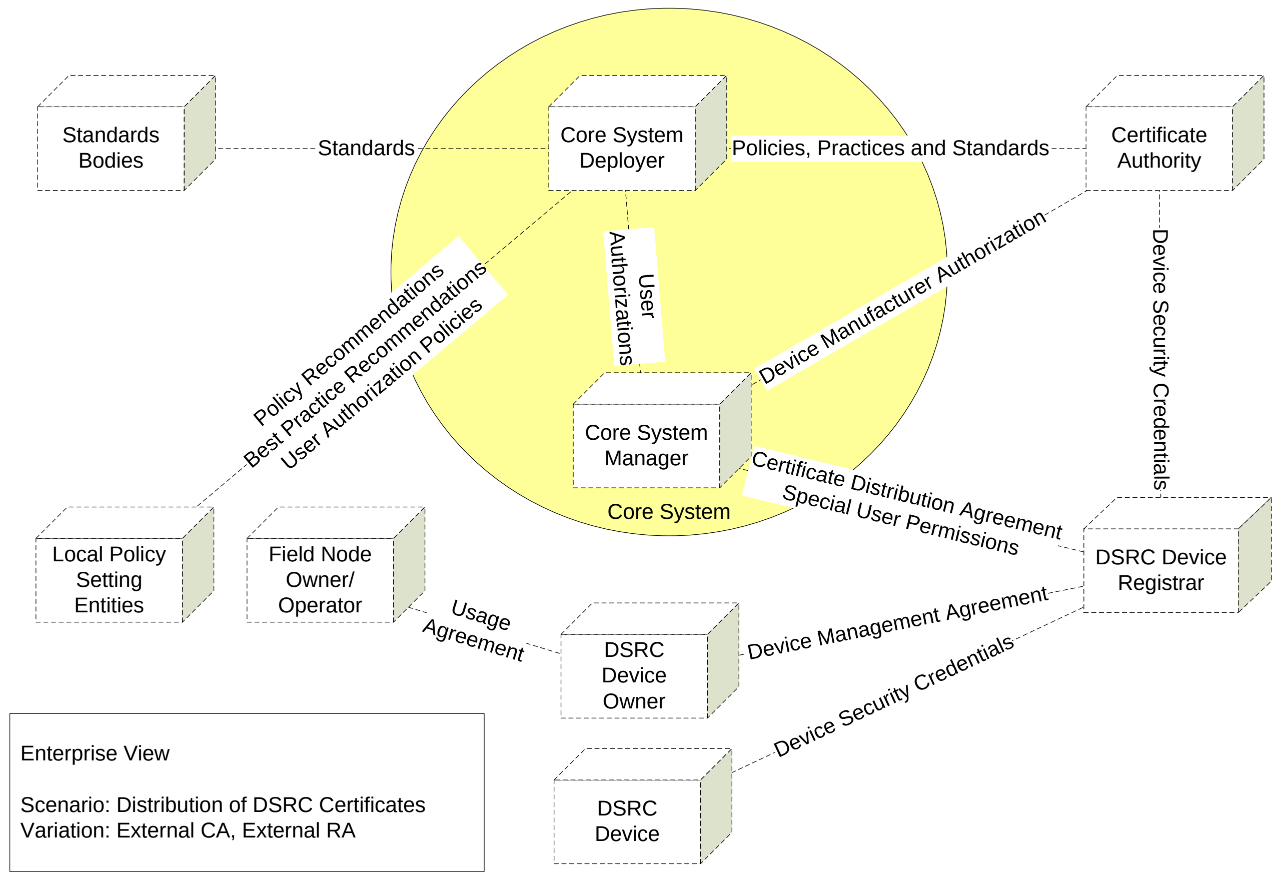

Standards Bodies
Standards
Core System Deployer
Policies, Practices and Standards
Certificate Authority
User Authorizations
Device Manufacturer Authorization
Device Security Credentials
Policy Recommendations
Best Practice Recommendations
User Authorization Policies
Core System Manager
Core System
Certificate Distribution Agreement
Special User Permissions
DSRC Device Registrar
Local Policy Setting Entities
Field Node Owner/ Operator
Usage Agreement
DSRC Device Owner
Device Management Agreement
Device Security Credentials
DSRC Device
Enterprise View
Scenario: Distribution of DSRC Certificates
Variation: External CA, External RA

Figure 4-1: Enterprise View - Security Credentials Configuration

4.1.1.4 Configuration Information

The following views must be considered when changing this view:

Functional Viewpoint: Functional View – DSRC Certificate Distribution.

4.1.1.5 Alternatives Considered

Three alternatives below explore different arrangements between CA and Registration Authority (RA). For a discussion of the technical roles of CA an RA, see Section 4.2.4 Functional View – DSRC Certificate Distribution.

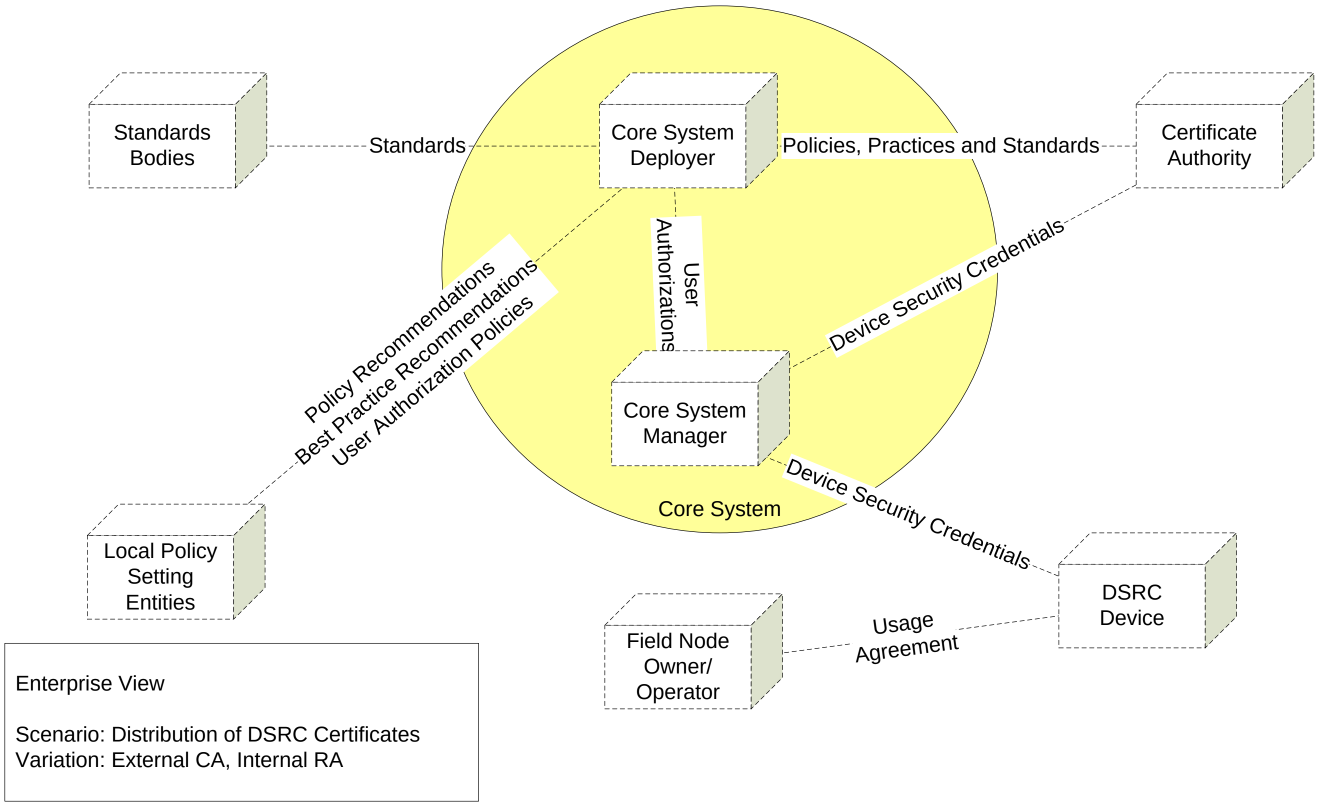

Standards Bodies
Standards
Core System Deployer
Policies, Practices and Standards
Certificate Authority
User Authorizations
Device Security Credentials
Policy Recommendations
Best Practice Recommendations
User Authorization Policies
Core System Manager
Core System
Device Security Credentials
Local Policy Setting Entities
DSRC Device
Field Node Owner/ Operator
Usage Agreement
Enterprise View
Scenario: Distribution of DSRC Certificates
Variation: External CA, Internal RA

Figure 4-2: Enterprise View - Security Credentials Configuration Alternative External CA, Internal RA

Figure 4-2: Enterprise View - Security Credentials Configuration Alternative External CA, Internal RA illustrates an alternative view where the CA functionality is provided by an external CA but registration is handled by the Core. This view is compatible with that shown in Figure 4-1 but not with that shown in Figure 4-3.

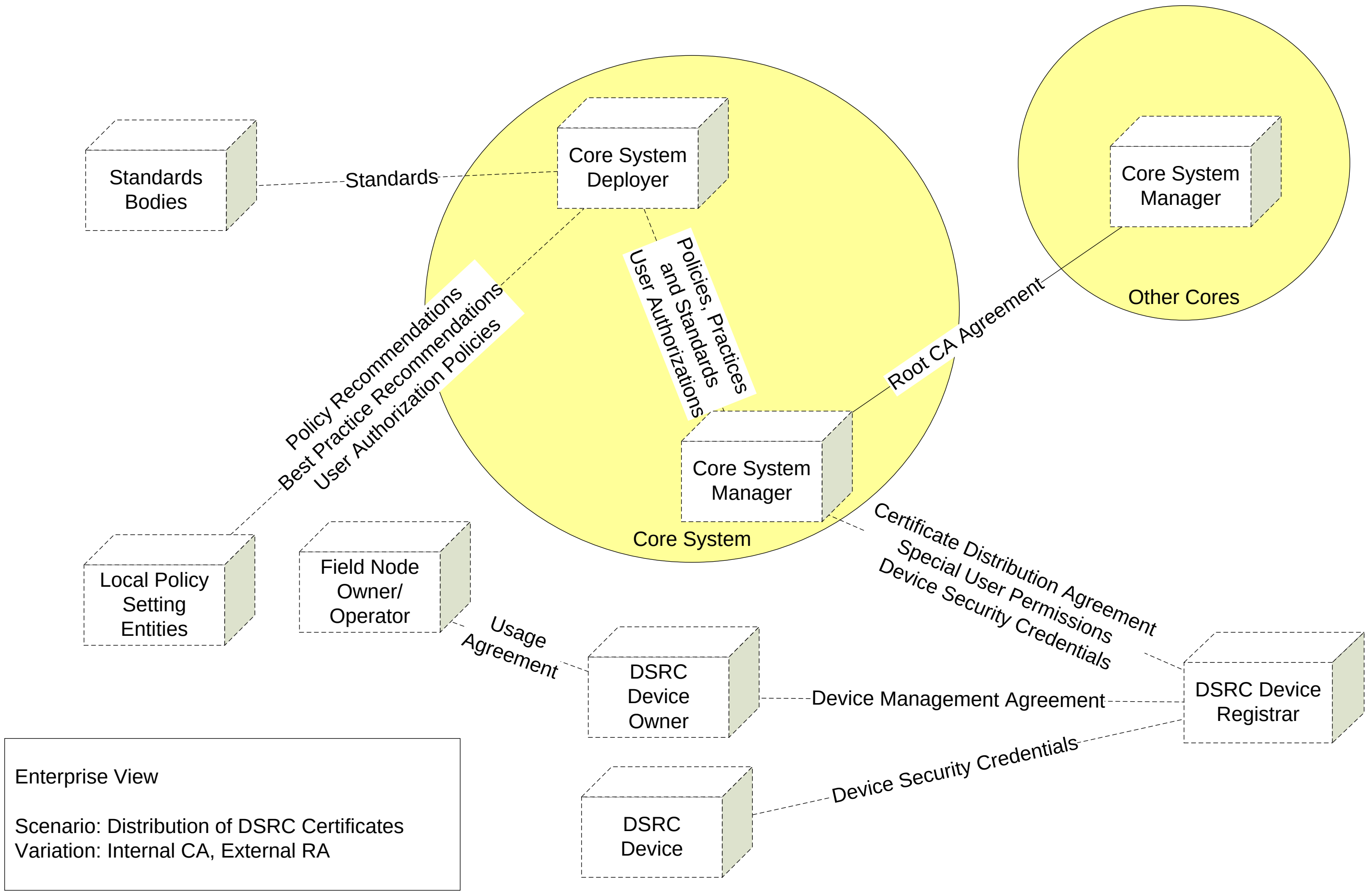

Standards Bodies
Standards
Core System Deployer
Policy Recommendations
Best Practice Recommendations
User Authorization Policies
Policies, Practices and Standards
User Authorizations
Core System Manager
Other Cores
Root CA Agreement
Core System Manager
Core System
Certificate Distribution Agreement
Special User Permissions
Device Security Credentials
Local Policy Setting Entities
Field Node Owner/ Operator
Usage Agreement
DSRC Device Owner
Device Management Agreement
DSRC Device Registrar
Device Security Credentials
DSRC Device
Enterprise View
Scenario: Distribution of DSRC Certificates
Variation: Internal CA, External RA

Figure 4-3: Enterprise View - Security Credentials Configuration Alternative Internal CA, external RA

Figure 4-3: Enterprise View - Security Credentials Configuration Alternative Internal CA, external RA illustrates an alternative view where the CA functionality is provided by the Core System but registration is provided by an external entity. In order for this to work, one Core must serve as the root CA, and be recognized by all other cores. This view is compatible with that illustrated in Figure 4-4, but not with Figure 4-1 or Figure 4-2.

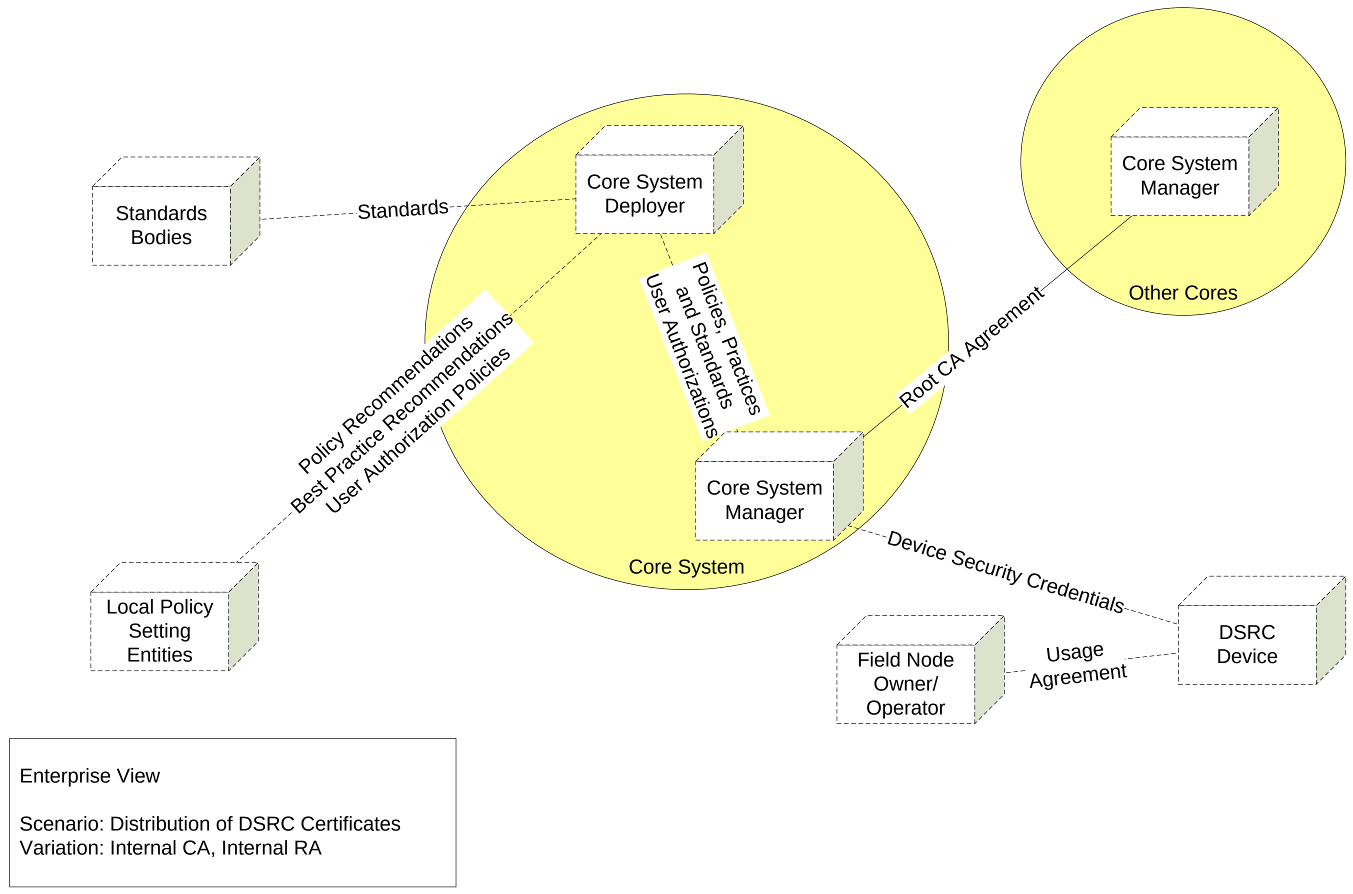

Standards Bodies
Standards
Core System Deployer
Other Cores
Core System Manager
Root CA Agreement
Policies, Practices and Standards
User Authorizations
Policy Recommendations
Best Practice Recommendations
User Authorization Policies
Core System Manager
Core System
Local Policy Setting Entities
Device Security Credentials
DSRC Device
Field Node Owner/ Operator
Usage Agreement
Enterprise View
Scenario: Distribution of DSRC Certificates
Variation: Internal CA, Internal RA

Figure 4-4: Enterprise View - Security Credentials Configuration Alternative Internal CA, Internal RA

Figure 4-4: Enterprise View - Security Credentials Configuration Alternative Internal CA, Internal RA illustrates an alternative view where the CA and RA functionality are both provided by the Core System. In order for this to work, one Core must serve as the root CA, and be recognized by all other cores. This view is compatible with that illustrated in Figure 4-3, but not with Figure 4-1 or Figure 4-2.

4.1.2 Enterprise View – Development

4.1.2.1 Introduction

The connected vehicle environment will require software development and distribution to implement Core Systems, and also to provide Core-enabled software at end user devices. This view illustrates the relationships between entities involved in developing, testing, certifying, specifying and deploying these various software components and applications.

4.1.2.2 Object Definitions and Roles

Application Certification Body: The enterprise charged with certifying applications for deployment.

Core Library Developer: The enterprise that develops library software for use on System User devices.

Core System Deployer: The Core System Deployer deploys the Core System. It acquires this system from the developer, for which it may enter into a funding arrangement, and provide constraints on system capabilities depending on the Deployer's scope.

Core System Developer: The Core System Developer is the enterprise that develops the Core System software and specifies its hardware configuration. It receives requirements and architecture from this SAD and the SyRS, develops detailed requirements and implements those requirements. It works with an independent test and validation enterprise (Core System Tester) to ensure that it has developed according to the requirements and architecture specified in these source documents. The Core System Developer may provide its configuration and software to a Core System Deployer, and may modify its software to take into account the Deployer's supplied constraints.

Core System Tester: The Core System Tester takes the USDOT's source documentation and uses that as the basis for verification and validation of Core System Developer's software.

End User: The Center, Field or Mobile User that wants to run software to interact with the Core System. This entity may enter into license agreements with application deployers to use their software.

End User Application Deployer: The enterprise that deploys applications onto End User devices. For Field and Center Users this would typically be a specialized contractor or software licensing organization. For Mobile Users this is more likely to be a service provider, e.g. a cellular service provider, automotive dealer.

End User Application Developer: The enterprise that develops applications targeted to End User devices.

End User System Deployer: This enterprise provides the physical devices and device configuration that End Users interact with. This could include an automotive manufacturer for OEM-supplied devices, a dealer or aftermarket shop for retrofit devices, a device manufacturer for aftermarket devices, a cellular service provider for smart phones, or a traffic signal controller manufacturer for DSRC-enabled signal controllers, etc.

Local Policy Setting Entities: This body imposes constraints on the deployment of the Core System according to local policies. Entities filling this role could include local DoTs, local, county and state government, regional transportation authorities.

Standards Bodies: Standards bodies develop standards and make those available to the Core System Deployer and Core Library developer. These necessary standards and their high level description will come from the USDOT's Core System documentation, including the ConOps, SyRS and this SAD, through the agency that funds the standard development.

Standards Funding Agencies: The entity or entities that provides funding and direction to standards development entities. This could be the USDOT or any combination of private and public agencies banded together under a separate agreement.

USDOT Core System Documents: The library of documents that make up the USDOT's specification for the Core System: The ConOps, SyRS and SAD.

4.1.2.3 View Description

This view addresses the enterprise relationships involved in the development and deployment of the Core System, and also the development and deployment of external applications.

There are four types of enterprises involved in this view: developers, deployers, end users, and support entities. Developers create and maintain end user applications, Core System software, and Core Library software[4]. Deployers install applications, library software, and Core software in operational configurations. Support entities provide foundational functionality, requirements, environments or capabilities used by the deployers or developers. End Users are the Mobile, Center, and Field Users that require the use of library and/or end user software.

This Enterprise View illustrates various relationships, mostly agreements or contracts, between Enterprises that are necessary to deploy functionality to Cores or End Users. External Application Developers and Deployers are included because they may have a relationship with the Core Manager that is relevant to them being able to provide applications to End Users. For example, the developer of a signal priority application may need the Core to ensure that the End Users of its application have the relevant permissions, which are encoded as part of their DSRC certificate. This requires the deployer of the signal priority application to work with the Manager of the Core to ensure those certificates include the requisite permissions.

Additionally, any certification of external applications would have to use the foundational documents describing the Core System to at least specify behavior that can affect the Core. The Application Certification Body uses standards to evaluate and certify applications; certification processes could be further affected by operational lessons learned; lessons learned come from the Core System Manager. A multitude of Cores may not have individual relationships with the Application Certifying Body; therefore lessons learned should be fed through the Core System Documentation and applied formally as notes or

[4] Core Library software is that software used by Users to access Core System functions. This is software resident on System User devices that interfaces with or uses data provided by the Core.

changes to the Core Requirements or Architecture, whereupon the resulting changes would be recognized as part of the application certification process.

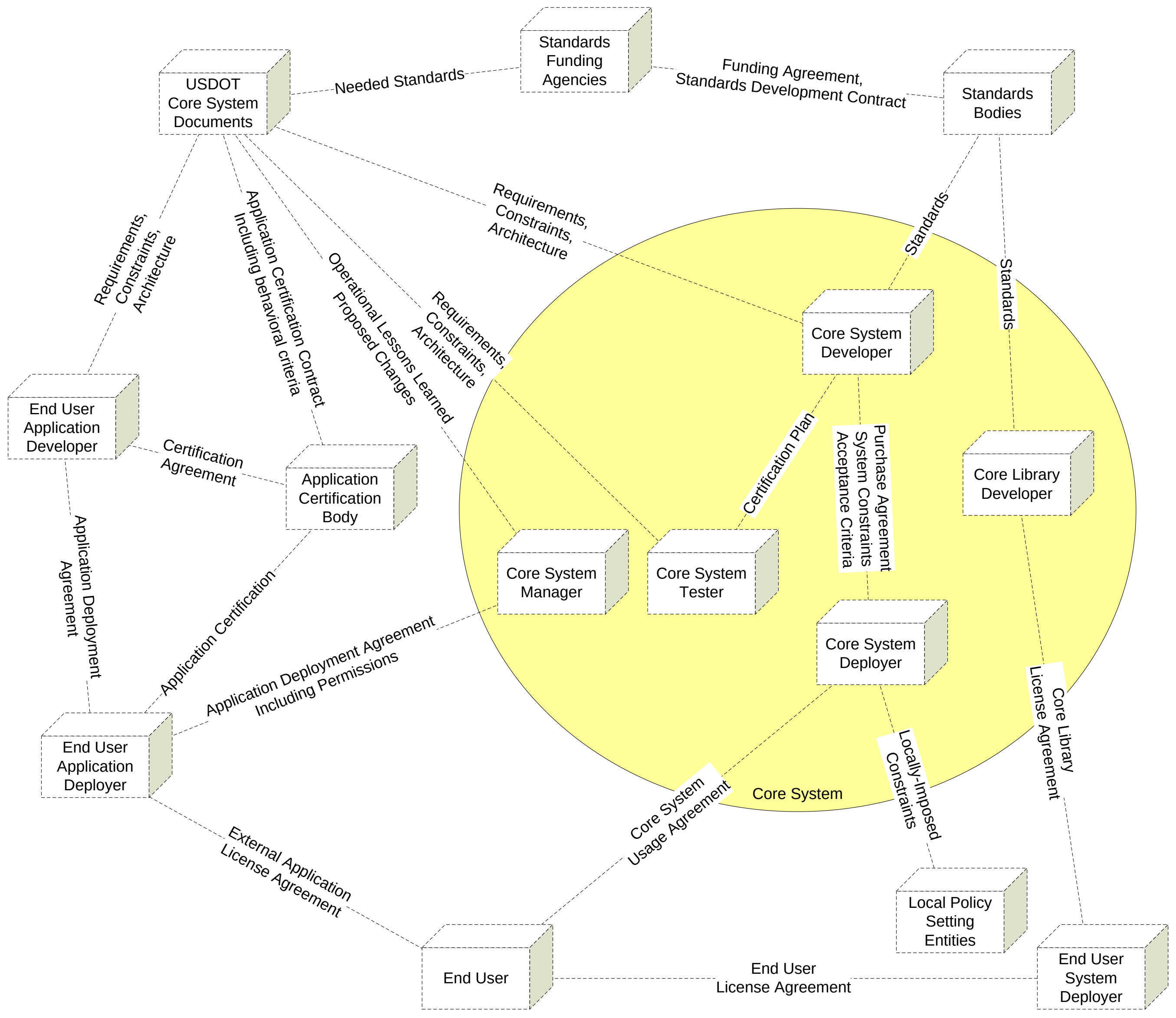

Standards Funding Agencies
USDOT Core System Documents
Needed Standards
Funding Agreement, Standards Development Contract
Standards Bodies
Requirements, Constraints, Architecture
Application Certification Contract Including behavioral criteria
Operational Lessons Learned Proposed Changes
Requirements, Constraints, Architecture
Requirements, Constraints, Architecture
Standards
Standards
End User Application Developer
Certification Agreement
Application Certification Body
Core System Developer
Certification Plan
Purchase Agreement System Constraints Acceptance Criteria
Core Library Developer
Application Deployment Agreement
Application Certification
Core System Manager
Core System Tester
Core System Deployer
Application Deployment Agreement Including Permissions
End User Application Deployer
Core System Usage Agreement
Core System
Locally-Imposed Constraints
Core Library License Agreement
External Application License Agreement
Local Policy Setting Entities
End User
End User License Agreement
End User System Deployer

Figure 4-5: Enterprise View - Development

4.1.2.4 Configuration Information

The following views must be considered when changing this view:

Enterprise Viewpoint: Enterprise View – DSRC Security Credentials Configuration
Enterprise Viewpoint: Enterprise View – Maintenance
Enterprise Viewpoint: Enterprise View – Governance

4.1.2.5 Alternatives Considered

If no applications require certification, then the Certification Body and its associated relationships can be ignored. This alternative is not illustrated, because it is a simple deletion of what exists above, and also because it is highly likely that some, at least mandatory applications, will require certification.

4.1.3 Enterprise View – Maintenance

4.1.3.1 Introduction

The Core System will require maintenance for software patches, configuration changes, hardware maintenance, and similar activities. This view explores the enterprise relationships necessary to perform those maintenance activities.

4.1.3.2 Object Definitions and Roles

Core System Deployer: The Core System Deployer deploys the Core System. It acquires this system from the developer, for which it may enter into a funding arrangement, and provide constraints on system capabilities depending on the Deployer's scope. It enters into a contractual relationship with the Core System Manager to operate the system, in which it includes desired policies and practices.

Core System Developer: The Core System Developer is the enterprise that develops the Core System software and specifies its hardware configuration. It works with an independent test and validation enterprise (Core System Tester) to ensure that it has developed according to the requirements and architecture specified in these source documents. The Core System Developer may provide its configuration and software to a Core System Deployer, and may modify its software to take into account the Deployer's supplied constraints. The Core System Developer trains both the Manager and Maintainer in how to operate and maintain the system, respectively.

Core System Maintainer: The Maintainer operates under a contract with the Core System Manager to maintain the Core. It receives training from the Developer, and Test Results of the original system configuration from the Core System Tester. The Maintainer should enter into an arrangement with the Field Node Owner/Operators to exchange maintenance information that is relevant to each.

Core System Manager: The Manager enters into a contract with the Deployer for the Manager to operate the Core System. The Manager enters into a contract with the Maintainer for the Maintainer to maintain the system. The Manager receives training in Core operations from the Developer.

Core System Tester: The Core System Tester provides the test results from the original system deployment to the Core System Maintainer, for the Maintainer to use as a baseline for system performance.

Field Node Owner/Operator: The owner and operator of the field infrastructure used by the DSRC Device Owner to the Core. This could be a cellular provider or DSRC field infrastructure manager. The Core Manager would want to enter into an agreement with this provider to share maintenance information, so that the Core System Manager can take maintenance and failure outages into account.

Local Policy Setting Entities: This body provides guidance on the operation of the Core System according to local policies and practices. Entities filling this role could include local DoTs, local, county and state government, regional transportation authorities, in particular the IT interests of those agencies.

4.1.3.3 View Description

This view addresses the enterprise relationships involved in the operations and maintenance of the Core System.

Most of the relationships are documented according to the Object Definitions and Roles above. Of note, there are only two entities outside the Core that have significant input to the operations and maintenance of the Core: Local Policy Setting Entities and the Field Node Owner Operator.

Policy Setters may impose their desires through law, funding incentive, or political pressure. This will vary from jurisdiction to jurisdiction as to the nature of the organizations involved. Maintaining consistency in this relationship will be difficult as the number of organizations filling this role could be large. This is a risk item and needs to be recognized by potential deployers. Without consistent approaches to IT best practices and management, Deployers (and in turn Managers and Maintainers) could spend significant resources accommodating these policies or even choosing not to deploy because of the difficulty of doing so.

The Field Node Owner/Operator comes in two very different shapes: the deployer of local DSRC infrastructure, and cellular network providers. The DSRC infrastructure owner/operator is likely to be tightly coupled to the Core System effort and sympathetic to the needs of the Core operators; DSRC infrastructure is after all focused on delivering Core services. The cellular provider will generally be a large corporation, for whom delivery of Core services is merely more data and potentially more subscribers. Establishing a working relationship between maintenance personnel at these corporations will be more problematic, and should be noted as another risk item. Without this relationship, Core Maintainers may not understand the communications environment they operator in, and thus encounter difficulty monitoring and debugging system performance issues.

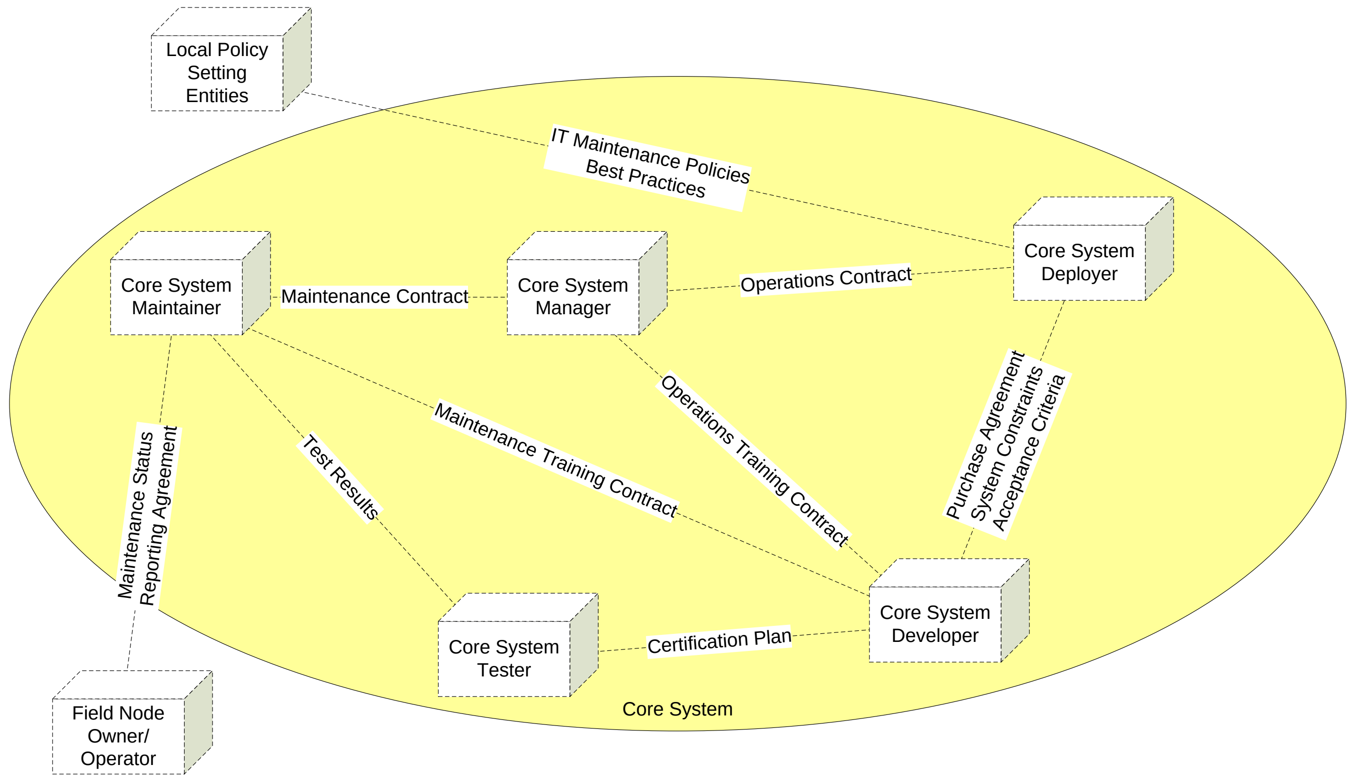


Figure 4-6: Enterprise View - Maintenance

4.1.3.4 Configuration Information

The following views must be considered when changing this view:

Functional View – System Monitor and Control

4.1.3.5 Alternatives Considered

None.

4.1.4 Enterprise View – Governance

4.1.4.1 Introduction

This view explores the enterprise relationships necessary to govern Core Systems through the exercise of management power to enforce policies.

4.1.4.2 Object Definitions and Roles

Core Certifying Body: The organization that verifies that a Core System adheres to the USDOT's specification. This includes the establishment of a Core Certification Plan, which identifies what must be done to certify that a Core meets the requirements and architecture defined in the SyRS and SAD. This plan is provided to the Core System Deployer prior to certification, and once certification takes place the results of that certification are also provided to the Deployer. Also, the Core System Manager is provided with the root certificate for their Core.

Once the Core is certified but prior to the Core beginning operations, the Core Certifying Body provides a Core Monitoring Plan to the Core System Manager. Once this occurs, the Core is permitted to operate. Periodic re-certification of the Core by this body will occur as specified in that Monitoring Plan.

The makeup of the Core Certifying Body is an unresolved policy question. At minimum, this body should include representatives of the following stakeholder groups:

- state/local DOTs
- US DOT including the Research and Innovative Technology Administration (RITA) Intelligent Transportation System (ITS) Joint Program Office (JPO), Federal Highway Administration (FHWA), Federal Transit Authority (FTA), National Highway Traffic Safety Administration (NHTSA), and the Office of the Secretary's (OST) Office of Policy
- Automakers
- Equipment vendors
- Mobile User telecommunications providers

Core System Deployer: The Core System Deployer deploys the Core System. It provides Core operating procedures, policies and configuration information to the Core System Manager, based on input received from Local Policy Setting Entities. This includes:

- What types of data to distribute as part of data distribution
- Who can subscribe to data, and what relationship that subscriber must establish with the Core in order to publish data
- Who can publish data using the Core, and what relationship that publisher must establish with the Core in order to publish data

The Deployer works with the Core Certifying Body to establish the Core Certification Plan, and receives the results of initial and subsequent certifications. The Deployer also establishes usage agreements with Field Node Owner/Operators that provide services within the Core's area of operations.

Core System Manager: The Manager receives initial and subsequent configuration information from the Core System Deployer, which provides the basis for setting user permissions and system configuration information. The Manager interacts with Data Publisher and Data Subscriber to exchange the information necessary to allow their data distribution actions, per the rules and procedures provided by the

Core System Deployer. The Manager works with the Core Certifying Body in the performance of re-certification according to the Monitoring Plan. The Core System Manager shares operational performance and scope information with other Cores, including what services the Core offers, to what body of users, and over what area those services are provided. The Core System Manager may also enter into an agreement with other Core System Managers to provide backup and/or service forwarding facilities.

Data Subscribers: Entities that wish to receive data through the Core may need to enter into an agreement with the Core, depending on the Core's operational policies.

Data Publishers: Entities that wish to provide data through the Core may need to enter into an agreement with the Core, depending on the Core's operational policies.

Field Node Owner Operator: This is the cellular or DSRC infrastructure provider that works with the Deployer to ensure Mobile Users have access to their infrastructure. This enterprise enters into a usage agreement with the Deployer, which describes what use the Core can make of the field node infrastructure and shares sufficient field configuration information to enable geo-broadcast functions. This includes updating such information when the configuration and/or locations of field nodes changes.

Local Policy Setting Entities: Organizations with legal or regulatory authority over the area that the Core operates may impose restrictions on procedure, data use and/or data provision. This information comes to the Core System Deployer, who uses it to establish system operational procedures and configuration, and is provided to the Manager for operations. Changes to these rules and policies follow the same path to the Manager.

External Support Systems: Systems outside the control of the Core that provide services critical to Core Operations (e.g., root X.509 CA). The Core Certifying Body would need to certify which external support systems are certified to provide services to a Core.

USDOT Core System Documents: The library of documents that make up the USDOT's specification for the Core System: The ConOps, SyRS, and SAD.

4.1.4.3 View Description

This view supports a federation of Core Systems that interoperate because they are certified against a common set of standards. The Core Certifying Body acts as the sole gatekeeper to operation of Core Systems. This implies that recognition of another Core is not possible without that Core being able to identify itself as certified by the Core Certifying Body. This in turn implies that the means for that identification, an X.509 digital certificate associated with that Core, cannot be granted without Core Certifying Body involvement. Thus the Core Certifying Body must be the controller of the X.509 certificate authority that distributes Core root certificates.

Other relationships are more straightforward. Core Systems may have relationships directly with one another to establish backup procedures, negotiate operational scope and related forwarding of service requests between Cores. Relationships between data providers and subscribers are between individual users Cores; if a subscriber or provider interacts with more than one Core, they will have a relationship with each Core.

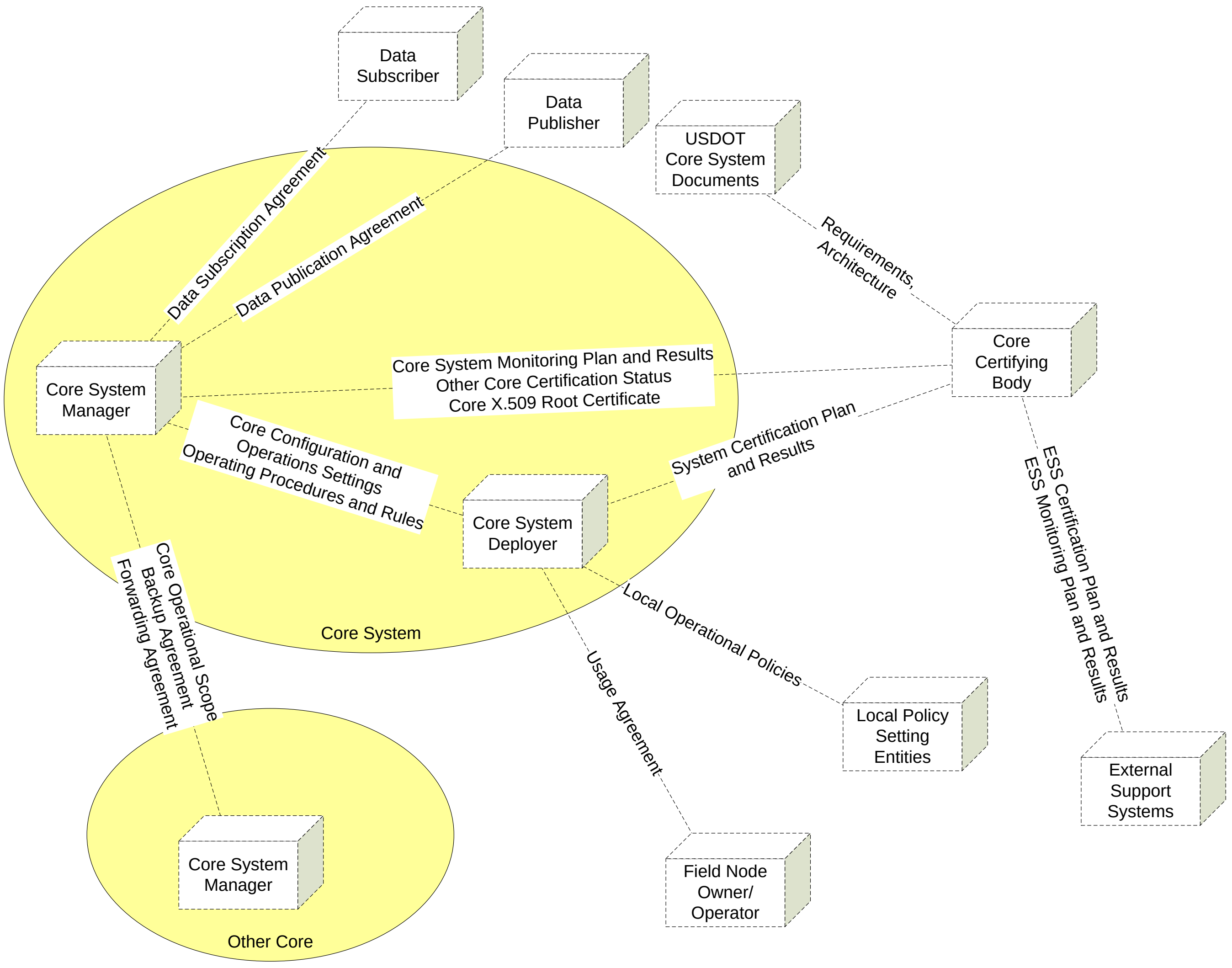

Data Subscriber
Data Publisher
USDOT Core System Documents
Data Subscription Agreement
Data Publication Agreement
Requirements, Architecture
Core Certifying Body
Core System Manager
Core System Monitoring Plan and Results
Other Core Certification Status
Core X.509 Root Certificate
Core Configuration and Operations Settings
Operating Procedures and Rules
System Certification Plan and Results
Core System Deployer
ESS Certification Plan and Results
ESS Monitoring Plan and Results
Core Operational Scope
Backup Agreement
Forwarding Agreement
Core System
Local Operational Policies
Usage Agreement
Local Policy Setting Entities
External Support Systems
Core System Manager
Other Core
Field Node Owner/ Operator

Figure 4-7: Enterprise View - Governance

4.1.4.4 Configuration Information

The following views must be considered when changing this view:

None.

4.1.4.5 Alternatives Considered

An alternative to the Core Certifying Body acting as gatekeeper to entry of Core Systems into operations is to simply allow the market to sort it all out. Cores could get their root certificates from a 3rd party provider (which should still be approved just to make sure it complied with the requirements in Core documents, though this is not strictly necessary since there are today several reputable X.509 CAs in operation).

This alternative relies on System Users and other Core Systems to provide the differentiating function between high performing Cores and lesser performing Cores. Cores that don’t perform well and don’t meet System Users' needs will not receive as much interaction with System Users, and thus be pressured to improve their operations. Similarly, those that are not good partners with other Cores won't have sharing or backup relationships.

This approach largely removes the need for the Core Certifying Body. It could even be completely eliminated. This is one less organization that would need to be formed and managed and paid for. It may make it easier for an entity that wants to operate a Core to deploy it, since it won't have to interact with the Core Certifying Body.

On the other hand, without the Core Certifying Body controlling entrance to the Core System market, there is no guarantee that Cores will operate the same way; they may not even be compliant with this architecture. This could jeopardize interoperability between Cores and thus the availability of digital certificates for Mobile Users. Mobile Users that cannot get digital certificates may not receive the same level of safety improvements in the *connected vehicle* environment as those that do have certificates.

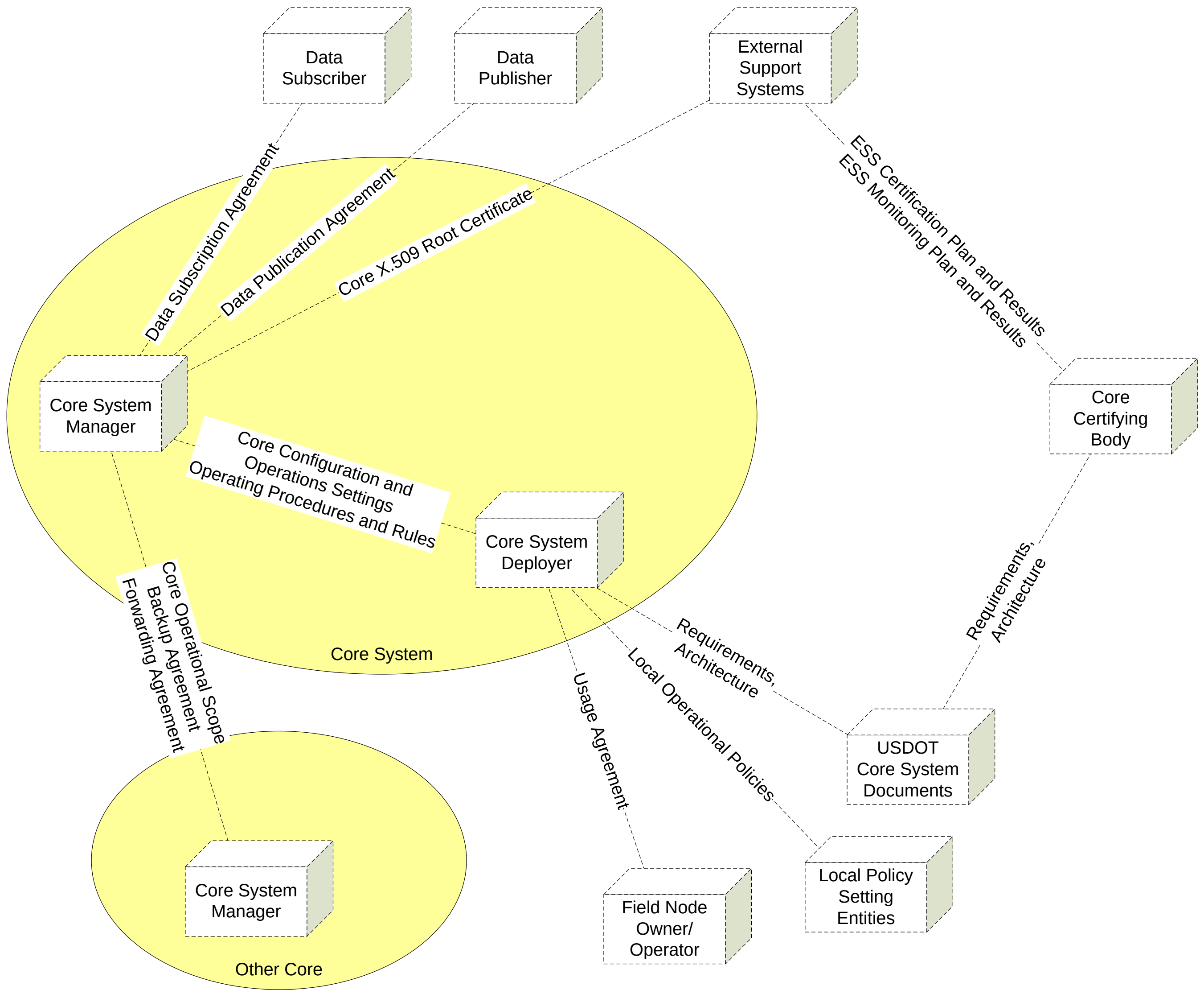
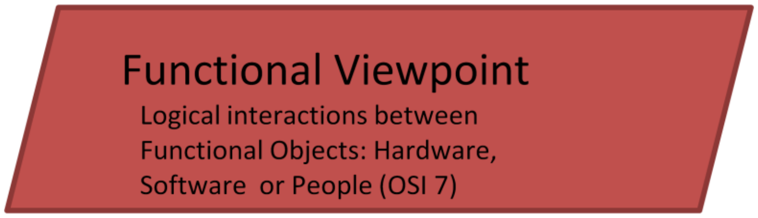

Data Subscriber
Data Publisher
External Support Systems
Data Subscription Agreement
Data Publication Agreement
Core X.509 Root Certificate
ESS Certification Plan and Results
ESS Monitoring Plan and Results
Core System Manager
Core Certifying Body
Core Configuration and Operations Settings
Operating Procedures and Rules
Core System Deployer
Requirements, Architecture
Core Operational Scope
Backup Agreement
Forwarding Agreement
Core System
Usage Agreement
Local Operational Policies
Requirements, Architecture
USDOT Core System Documents
Core System Manager
Other Core
Field Node Owner/ Operator
Local Policy Setting Entities

Figure 4-8: Enterprise View - Governance Market Alternative

4.2 Functional Viewpoint


Functional Viewpoint
Logical interactions between Functional Objects: Hardware, Software or People (OSI 7)

The first functional view starts with the subsystems defined in the Concept of Operations. Subsequent views explore the functions associated with a particular scenario (e.g., Data Distribution). Scenario-based views were chosen to illustrate functions where significant interactions take place between subsystems. Future versions of this document may include subsystem-based views, showing only those functions in a particular subsystem and their interfaces.

Scenario-based functional views are presented for six aspects of Core System operation: distribution of data (primarily through a publish-subscribe mechanism), management of Core operations, management and distribution of 1609.2 certificates, management and distribution of X.509 certificates, decryption of encrypted messages, and provision of Internet connectivity.

Many functions appear in more than one view. A function that appears in multiple views always provides the same function, but may interact with different functions in different views. Unless otherwise stated, all views are equally valid. For example, function A may appear in two different views. In one view it interacts with functions B and C, and in another view it interacts with C, D and E. Unless one of these views is identified as an alternative, both views are equally valid and thus function A interacts with B, C, D and E.

Functional view diagrams use a color coding convention to identify the subsystem a function belongs to, as illustrated in Figure 4-9.

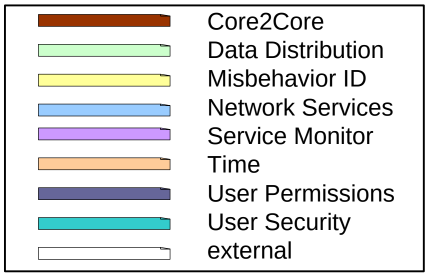


Figure 4-9. Functional View Subsystem Color Codes

Some of these views refer to the type or identity of System Users. When referring to the identity of a System User, the Core System is indicating unique information attributable to a specific individual person. When referring to System User type, the Core is indicating unique information attributable to a group of individuals that share characteristics but are not necessarily uniquely identifiable. For example, User_ID 1234 may refer to John Smith, a unique user of the Core, and Type_ID Bus may refer to System Users that are all buses.

Types are not defined in this document, with one exception: Anonymous. Anonymous users are System Users that do not provide their identity when they communicate with the Core System. Anonymous users may be required to provide their identity in order to initially obtain digital certificates if the Core is acting as Registration Authority (see 4.1.1 Enterprise View – DSRC Security Credentials Configuration and 4.2.4 Functional View – DSRC Certificate Distribution), but not subsequently, nor at any other time when communicating with the Core.

Some Functional Objects operate differently depending on the Core System state or their subsystem's operational mode. As described in the Concept of Operations, the Core may be in one of five states: Installation, Standby, Training, Operational, and Maintenance. Each subsystem may be in one of four modes: Normal, Restricted, Degraded, or Degraded Restricted. Only the Operational state is considered in these functional views at this time (a future version of this document will consider the other states). Some Functional Objects explicitly function differently when in one of the two Restricted modes; this is noted in the definitions of those Functional Objects.

4.2.1 Functional View – Top Level

4.2.1.1 Introduction

This view documents the top level functional view of the Core System. The objects in this view map to the subsystems in the Concept of Operations, and provide the basis for all subsequent functional views.

4.2.1.2 Object Definitions and Roles

Core2Core interfaces with other Core Systems, advertising its jurisdictional scope, offered services, and services it desires from other Cores. Core2Core maintains a knowledge base of services available from other Cores including other Cores' jurisdictional scope.
Additionally Core2Core is responsible for compatibility between Cores, ensuring that it does not encroach on the scope of another Core, and similarly accepting error messages from Mobile Users that might indicate a cross-jurisdictional compatibility or scope coverage issue.

Associated Information Objects:

All Core2Core objects are shared between Cores.

- **Misbehaving User Info** contains a certificate ID, time and type of misbehavior identified; this information is shared between Cores so that evidence of misbehavior across Core boundaries can be combined to determine bad actors that might not be identified by individual Cores.
- **Distributed Certificate IDs** is a list of the certificate IDs distributed or activated by a Core.
- **Certificate Revocation List** is the list of active certificates that are no longer valid.
- **Permission Revocation Info** describes the permissions that have been removed from a given System User.
- **Certificate Request Broker Info** is a proxied request for services originating from a System User. When a System User requests service from a Core that does not provide them, the Core may forward that request to another Core that does provide those services.

Data Distribution maintains a directory of System Users that want data and facilitates the delivery of that data to those users. It supports two distribution mechanisms:

- Source-to-Points: The data provider communicates data directly to data consumers. In this case, no data is sent to the Core System, however the Core is involved to check User Permissions and provide addressing services through those subsystems
- Publish-Subscribe: The data provider communicates data to the Core, which forwards it to all users that are subscribed to receive the data.

Data Distribution maintains a registry of which data consumers get what data, according to the criteria defined above. Data Distribution does not store or buffer data beyond that which is necessary to complete publish-subscribe actions. If a given data consumer is unable to receive data that it has subscribed to because of a communications or other system failure, the data in question may be lost.

Data Distribution repackages data it receives from data providers, stripping away the source header information while maintaining the message payload. It then sends the repackaged payload data to subscribers of that data.

Associated Information Objects:

- **Publication Request** is a request from a System User to publish data using the Core's Data Distribution functions. It includes the type of data the System User wishes to publish, and the expected frequency with which the System User will provide that data.
- **Subscription Request** is a request from a System User to receive data from the Core that is being published by other users, and distributed through the Core. It includes criteria including data type, data quality characteristics, data format requirements, geographic area, sampling rate, minimum and maximum frequency of data forwarding.
- **Publication Confirmation** is the Core's response to a prospective data provider, information them whether or not they can publish data using the Core, and what data they can publish. In the case where the System User is attempting to publish data that the Core does not handle through publish-subscribe but another external entity does, this message includes the contact information for that 3rd party data sink.
- **Subscription Confirmation** is the Core's response to a prospective data subscriber, information them whether or not they are subscribed to data from the Core, and what data they are subscribed to.
- **Data** is the data received by the Core from publishers, or sent by the Core to subscribers, as part of the publish/subscribe action.

Misbehavior Management analyzes messages sent to the Core System to identify users operating outside of their assigned permissions. It works with the User Permissions subsystem to identify suspicious requests and to maintain a record of specifically identifiable users that:

1. Provide false or misleading data
2. Operate in such a fashion as to impede other users
3. Operate outside of their authorized scope

Because most end users will rarely interface with the Core System, the Core will also accept reports of misbehaving users from System Users. System Users can send misbehavior reports that reference credentials attached to messages, and note the type of misbehavior in question. Misbehavior Management will record such reports, and according to a set of rules, determine when to revoke credentials from such reported misbehaving users. For anonymous users revocation is more complex, and may result instead in a lack of credential renewal.

Associated Information Objects:

- **Misbehavior Reports** a report of misbehavior received from System Users. It includes an identification of the reporting System User, ID or pseudo-ID of the misbehaving user, and characterization of the type of misbehavior.

Network Services provides information to System Users that enables those users to communicate with a group of System Users, by maintaining information regarding available communications methods, coverage areas, addresses and performance characteristics for geo-cast communications.

Network Services also provides management for communications resources. It provides the Core System's Internet access and network security, including port monitoring and firewall functionality. It enables decisions about which communications medium to use when more than one is available. This includes identifying available communications methods current performance characteristics and applicable user permission levels.

Associated Information Objects:

- **Geo-broadcast Request** is request from a System User for the information that user needs in order to distributed data over a defined geographic area. This will include at minimum a description of the area over which the System User wants to distribute data.
- **Geo-broadcast Info** is a response from the Core informing the System User of the information necessary to perform a geo-graphic broadcast.

Service Monitor monitors the status of Core System services, interfaces, and communications networks connected to the Core. It informs System Users of the availability and status of Core services.

Service Monitor also monitors the integrity of internal Core System components and supporting software and mitigates against vulnerabilities. This includes periodic verification of the authenticity of Core service software and supporting software. This also includes monitoring for vulnerabilities including but not limited to virus protection and monitoring for patches to third party components. Should vulnerability be detected, or a component of the Core found to have lost integrity, Service Monitor takes steps to mitigate against damage and performance degradation.

Service Monitor monitors the environmental conditions that Core components operate in (temperature and humidity) as well as the condition of its power system. It takes steps to mitigate against system failures in the event that environmental conditions exceed operating thresholds.

Service Monitor also monitors the performance of all services and interfaces and makes performance metrics available to Core Operators.

Associated Information Objects:

- **Service Status** describes the status of all Core subsystems, including state and mode.
- **Service Status Request** is a request from System Users for the status of Core subsystems.

Time uses a time base available to all System Users and makes this time available to all Core System services which use this time base whenever a time reference is required.

Associated Information Objects:

- **Time** is time information received from a standardized, available time source.

User Permissions provides tools allowing Operators and other Core System components to verify whether a given user, identified by digital certificate-based credentials, is authorized to request or perform the action requested in the message payload. It also maintains the status of users, whether they have a specific account or belong to an anonymous group, their allowed behaviors with defined permissions (publish, subscribe, actions allowed to request, administrate, etc.). User Permissions provides tools for Operators to create new users and groups, modify existing users and groups, and modify permissions associated with users and groups.

Associated Information Objects:

- **User Permission Confirmation** is the Core's response to a System User asking for permission to use the Core's services. This will include the System User's ID and description of allowed interactions with the Core. If the System User requested special permissions managed by Core-provided certificates, the status of that request will be included as well.

- **User Identity and Permission Request** is a submission of identity credentials and a request for permissions granted to use Core System services and potentially, application services managed by certificates distributed by the Core.

User Security manages access rules and credentials in the form of digital certificates, including X.509, 1609.2 identity and 1609.2 anonymous certificates, for all System Users and Core System components that require and are entitled to them. It creates and distributes cryptographic keys to qualifying System Users. It works with User Permissions to determine whether a given user applying for credentials or keys is entitled to them. It also manages revocation of credentials, distributing Certificate Revocation Lists (CRLs) of obsolete or disallowed credentials to interested System Users. User Security may use an External Support System to manage certificates; information describing this approach depicted in succeeding architecture views.

Associated Information Objects:

- **Certificate Request** is a request for a new certificate, set of certificates, or activation of existing certificates from a System User. This request could apply to 1609.2 anonymous or identity certificates, or X.509 certificates.
- **Certificates** are the activated 1609.2 or X.509 certificates or activation codes for existing certificates the Core provides in response to a certificate request.
- **Certificate Revocation List** is the list of active certificates that are no longer valid.

4.2.1.3 View Description

This view illustrates the highest level functions the Core System provides. The objects defined above map directly to the subsystems defined in section 5 of the Concept of Operations. The object definitions provide most of the descriptive information for this view.

The view image does specify which subsystems provide external interfaces, and the high-level view of the types of information provided, received or exchanged across those external interfaces. Internal interfaces are indicated, but the information that flows between them is left for lower level diagrams. Every subsystem interacts with every other subsystem in some way; subsequent views explore these interactions in more detail.

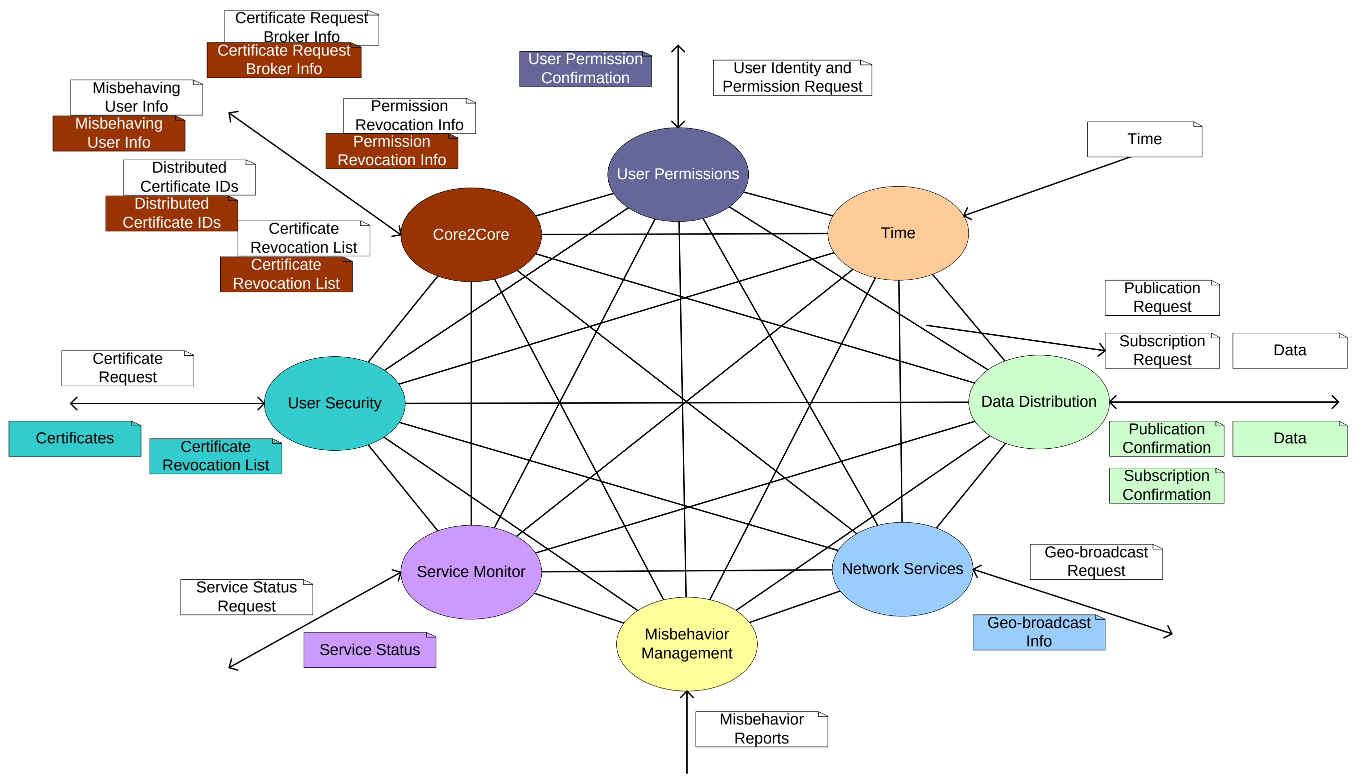


Figure 4-10: Functional View - High Level

4.2.1.4 Configuration Information

The following views must be considered when changing this view:

Functional Viewpoint: Functional View – Data Distribution
Functional Viewpoint: Functional View – System Monitor and Control
Functional Viewpoint: Functional View – DSRC Certificate Distribution
Functional Viewpoint: Functional View – X.509 Certificate Management
Functional Viewpoint: Functional View – Core Decryption
Functional Viewpoint: Functional View – Networking
Connectivity Viewpoint Connectivity View – High Level
Connectivity View – Core System Function Allocation

4.2.1.5 Alternatives Considered

None.

4.2.2 Functional View – Data Distribution

4.2.2.1 Introduction

This view explores the Core System's Data Distribution functionality. System Users provide data, other System Users subscribe to data; the Core matches those providers and consumers without requiring them to enter into a relationship with the other.

4.2.2.2 Object Definitions and Roles

Special Objects: These four objects interact with various other functions to perform operations on messages sent or received. They are not shown interacting on the diagram for the sake of clarity.

- **Decrypt Messages Received Encrypted:** This function accepts an encrypted message and decrypts it using the Core's private key.
- **Verify Signature of Received Messages:** This function verifies that the signature attached to a message is legitimate using the sender's public key.
- **Sign Messages:** This function attaches the Core System's digital signature to a message by encrypting a hash of the message content with the Core's private key.
- **Encrypt Messages:** This function encrypts a message using the public key of the intended recipient.

Check User Permission: This function accepts a System User ID or Operator ID, along with a type of operation that the user is attempting to access, and responds with whether or not the user is permitted that action.

Associated Information Objects:

- **User ID, function** is the identity of the System User attempting to perform a **function**, which is sent to the Check User Permission function to verify whether the System User is permitted to perform that **function**.
- **Subscribing ID, function** is the identity of the System User data subscriber, sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Provider ID, function** is the identity or type of the System User data provider (which could be anonymous), which is sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Operator ID, function** is the identity of the operator, which is sent to the Check User Permission function to verify whether the operator is permitted a given **function**.
- **Permission** is the response from the Check User Permission function describing whether or not the **User** is permitted this action.

Coordinate Data Coverage Area with Other Cores: Exchanges data with other Cores describing data distribution characteristics used by those Cores.

Associated Information Objects:

- **Data Coverage Characteristics** are exchanged with the Maintain Data Acceptance Catalog, and also with other Cores. This describes the boundaries of the data acceptance criteria used by the reporting Core. It includes data types, source, geographic area and time of data distribution.

- **Data Coverage Query** asks the Maintain Data Acceptance Catalog what the Core's data coverage characteristics are.
- **Data Coverage Characteristics Response** is the query response from the Maintain Data Acceptance Catalog that describes the boundaries of the data acceptance criteria used by the reporting Core. It includes data types, source, geographic area and time of data distribution.

Distribute Direct Data Acceptance Information: This function provides the information necessary for a System User to provide his data directly to a user of that data, without the Core's data distribution function being engaged.

Associated Information Objects:

- **Direct Data Distribution Info** includes the IP address and format expectations of the data sink.
- **Data Acceptance Query** is a request for information concerning direct data consumers. Includes the type of data to be provided, source type and location.

Identify Potentially Misbehaving Data Provider: Examines data to determine if its source is misbehaving; it looks for data that is logically inconsistent or has characteristics of malicious intent.

Associated Information Objects:

- **Data** that comes from the Receive Data from System Users function.
- **Misbehavior Report** is generated for malicious or illogical data.

Identify Potentially Misbehaving Data Provider: Examines subscription requests that are improperly formatted or that originated from a System User that is not permitted to subscribe to the data requested in the subscription request, and determines if the System User is misbehaving.

Associated Information Objects:

- **Misbehavior Report** is generated for a pattern of misbehavior associated with Suspicious Data Subscription Requests.
- **Suspicious Data Subscription Request** is a subscription request that is either improperly formatted or asks for data that the requesting System User is not permitted to receive; this is received from the Modify System User Data Subscriptions function.

Maintain Data Acceptance Catalog: This function maintains a catalog of data types and sources that are used by the Core's data distribution function. It allows changes to the catalog to update existing and add new data types, sources and combinations of types and sources. This function responds to queries about data types telling whether or not that type/source combination is accepted for data distribution. It also analyzes its data distribution coverage area, time, and source/type combinations against those reported by other Cores to determine if there are any overlaps (where multiple Cores provide the same services in the same area) or conflicts (where Cores provide different and conflicting information for the same area, for instance if one Core says that a given data type must go to external sink A, but another Core says that it accepts that data type).

Associated Information Objects:

- **Data Types and Sources** sent to the Maintain Data Acceptance Catalog function by the Repackage Data function as a query. This includes the types of data and their source or source type.

- **Data Acceptance Changes** received from the Modify Data Acceptance Catalogs function. This includes the types of data and their source or source type, and whether the Core should accept or not accept that data for distribution.
- **Data Acceptance or Discard** for each data type/source combination included in Data Types and Sources this includes a disposition, either forward or discard.
- **Data Type and Source** sent to the User Permissions function. This includes the types of data and their source or source type.
- **Permission** is the response from the Check User Permission function describing whether or not the **Data Type and Source** is permitted to provide the given type of data.
- **Data Type and Source Request** sent to the Maintain Data Acceptance Catalog function. This includes the types of data and their source or source type.
- **Existing Acceptance and/or Changes** sent to the Maintain System User Data Subscriptions function. This response to Data Type and Source Request indicates what changes (if any) were made to the data acceptance catalog, and whether or not the data type/source are already being distributed or if this is a new distribution.
- **Data Coverage Query** is the query from the Coordinate Data Coverage Area with Other Cores function asking the Maintain Data Acceptance Catalog what the Core's data coverage characteristics are.
- **Data Coverage Characteristics Response** is the query response to the Coordinate Data Coverage Area with other Cores function that describes the boundaries of the data acceptance criteria used by the reporting Core. Includes data types, source, geographic area and time of data distribution.
- **Data Coverage Conflict** is a notification sent to the Provide Operator Interface to DD function indicating a spatial, temporal or source/type combination overlap in data collection between this Core and another Core.
- **Data Acceptance Details** is information describing what data (source/type, time, coverage area) the Core accepts, sent to the Provide Operator Interface to DD function.

Maintain Direct Data Acceptance Catalog: Maintains a catalog of data consumers that do not use the Core's data distribution function to acquire data. Allows changes to the catalog to update existing and add new consumers.

Associated Information Objects:

- **Data Acceptance Changes** are modifications to be made to the data acceptance catalog, received from the Modify Data Acceptance Catalogs function.
- **Data Acceptance Query** is a request for information concerning direct data consumers. Includes the type of data to be provided, source type and location.
- **Direct Data Distribution Info** includes the IP address and format expectations of the data sink.

Maintain Geo-cast Information: Maintains a catalog of DSRC field device information, including IP addresses cross-referenced with locations, ranges, types and other pertinent information, to facilitate geographic-focused broadcast.

Associated Information Objects:

- **Geo-cast data request** is the query sent from a System User to the Core, asking for information required to provide data to a specified geographic area. It includes a specification of that area and the identification of the System User.

- **Geo-cast facilitation information** is the response to the query that describes what is required to geo-cast over the requested area.

Maintain System User Data Subscriptions: Maintains a catalog of data types and sources and the System Users that want to receive that data. Allows changes to the catalog to update existing and add new subscribers. Responds to queries about data subscribers and their subscriptions.

Associated Information Objects:

- **Data Type and Source Request** is a query sent to the Maintain Data Acceptance Catalog function. This includes the types of data and their source or source type.
- **Existing Acceptance and/or Changes** is the response to the Data Type and Source Request query from the Maintain Data Acceptance Catalog function. It specifies whether the data type/source in question is already being distributed or if not, that such distribution has been added.
- **Data Description** is the query received from the Match Data to Data Subscribers function, identifying the data type/source for which that function needs to know who is subscribed.
- **Subscription Details** is the response to the Data Description query sent to the Match Data to Data Subscribers function. It includes the identity/type of the subscribers matched to the data from the Data Description query, along with the subscriber's communications information.
- **Subscriber ID** is a query from the Modify System User Data Subscriptions function identifying a data subscriber, requesting the subscriber's existing data subscription information.
- **Data Subscription Details** is the query response to Subscriber ID or Data Subscription Changes sent to the Modify System User Data Subscriptions function. This includes the subscriber's ID and complete data subscription description.
- **Data Subscription Changes** are the changes to a subscriber's subscription sent by the Modify System User Data Subscriptions function.

Match Data to Data Subscribers: Examines repackaged data to determine what data goes to which subscribers. Adds subscriber destination information to data with subscriptions and passes it along to be distributed.

Associated Information Objects:

- **Repackaged Data** is composed of groups of same-type data, received from the Repackage Data function.
- **Repackaged Data with Destinations** is the **Repackaged Data** with the IP addresses of subscribers to the data attached.
- **Data Description** is a query sent to the Maintain User Data Subscriptions function, identifying the data type/source.
- **Subscription Details** is the response to the Data Description query. It includes the identity/type of the subscribers matched to the data from the Data Description query, along with the subscriber's IP address.

Modify Data Acceptance Catalogs: Modifies the Data Acceptance and Direct Data Acceptance Catalogs, indicating what types/sources of data the Core accepts, and what types are handled directly by external data sinks.

Associated Information Objects:

- **Data Acceptance Changes** are modifications to be made to the direct data acceptance catalog, sent to the Maintain Direct Data Acceptance Catalog function.
- **Data Acceptance Query** is a query for information concerning direct data consumers sent to the Maintain Direct Data Acceptance Catalog function. Includes the type of data to be provided, source type and location.
- **Direct Data Distribution Info** includes the IP address and format expectations of the data sink.
- **Data Acceptance Changes** sent to the Maintain Data Acceptance Catalog function. This includes the types of data and their source or source type, and whether the Core should accept or not accept that data for distribution.

Modify Operational State: is a controlling function that instructs various sub-functions of Data Distribution to change the way they operate. In restricted mode, Receive Data from System Users will be instructed to stop receiving data except from specified users, and Provide Data to Subscribing users similarly restricted to provide data only to specified users.

Associated Information Objects:

- **Operational Changes** describe how the Receive/Provide functions need to alter operations. If in a fully operational state this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to fully operational mode.

Modify System User Data Subscriptions: Modifies existing or creates new System User data subscriptions. Accepts input from System Users requesting new or modified subscriptions. Also accepts Operator input for creation of new and modification of existing subscriptions. Queries Check User Permissions to ensure that the System User in question is permitted to subscribe to the data he asks for, and to ensure that the entity requesting the change (either System User or Operator) is permitted to do so.

Associated Information Objects:

- **Data Subscription Request:** A request for a data subscription from a System User. It includes the user's identification and the data types, sources and time of data collection.
- **Data Subscription Details** sent to the Maintain Data Acceptance Catalog function. This includes the types of data and their source or source type.
- **Data Subscription Changes** is a request sent to the Maintain System User Data Subscriptions function. It specifies a System User and changes (additions if new) to the user’s subscription.
- **Permission** is the response from the Check User Permission function describing whether or not the **User** is permitted this action.
- **Subscriber ID, function** is the identity of the System User attempting to subscribe or have his subscription modified, sent to the Check User Permission function to verify whether the System User is permitted a given **function**. In this case the check would be to ensure that the user attempting the modification was allowed to make the subscription modification, and also to check that the subscriber was permitted to subscribe to the data it asks for.
- **Suspicious Data Subscription Request** is a subscription request that is either improperly formatted or asks for data that the requesting System User is not permitted to receive.

This request is forwarded to the Identify Potentially Misbehaving Data Subscriber function for analysis.

Provide Data to Subscribing System Users: Receives blocks of same type/source packaged data with destination information attached. It also buffers data packages intended for the same destination and sends those as larger messages. If operating in a restricted mode this function will prioritize data provision based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Repackaged, Addressed Data** is the packaged data on subscribed data types sent to data subscribers.
- **Repackaged Data w/Destinations** received from the Match Data to Data Subscribers function is a same type/source data block with a list of IP addresses that are to receive that data.

Provide Geo-Cast Information to System User: Responds to a query from a System User providing the System User with the information necessary to provide information to a specified geographic area. If the System User is not permitted this action, no response will be given.

Associated Information Objects:

- **User ID, function** is the identity of the System User querying for geographic information distribution and is sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Geo-cast data request** is the query sent from a System User to the Core System asking for information required to provide data to a specified geographic area. It includes a specification of that area and the identification of the System User.
- **Geo-cast facilitation information** is the response to the query that describes what is required to geo-cast over the requested area.
- **Permission** is the response from the Check User Permission function describing whether or not the **User** is permitted this action.

Provide Operator Interface to DD: Provides an interface to the Operator, allowing him access to User Data Subscription, data acceptance and geographic broadcast functions. This functional view is only concerned with direct access to geo-broadcast and data distribution functions, and not control of operational states or modes which is covered under a different view.

Associated Information Objects:

- **Data Acceptance Details** is information describing what data (source/type, time, coverage area) the Core accepts, received from the Maintain Data Acceptance Catalog.
- **Data Coverage Conflict** is a notification received from the Maintain Data Acceptance Catalog function indicating a spatial, temporal or source/type combination overlap in data collection between this Core and another Core.
- **Operator ID, function** is the identity of the Operator attempting a data subscription or data acceptance modification, sent to the Check User Permission function to verify whether the System User is permitted the given **function**.
- **Permission** sent to the Provide Operator Interface to DD function. This includes whether or not the Operator is permitted the **function** requested.

Receive Data from System Users: Acquires data sent by System Users. If operating in a restricted mode, this function will prioritize data acquisition based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Data that** comes from System Users and is distributed to the Repackage Data and Identify Potentially Misbehaving Data Provider functions

Repackage Data: Breaks data into component pieces; queries Maintain Data Acceptance Catalog function to determine what data has subscribers. Strips source header from data, and for each data piece provides those component pieces that are to be used to the Match Data to Data Subscribers function.

Associated Information Objects:

- **Data** that comes from Receive Data from System Users Function.
- **Repackaged Data** is composed of groups of same-type data, send to the Match Data to Data Subscribers function.
- **Data Types and Sources** sent to the Maintain Data Acceptance Catalog function. This includes the types of data and their source or source type.
- **Data Acceptance or Discard:** for each data type/source combination included in **Data Types and Sources**, includes disposition: either forward or discard.

4.2.2.3 View Description

This view addresses the functions required to implement, monitor and control the Core System's data distribution function. This includes:

- Responding to System User's data provision requests, informing them if the Core accepts the data they have to provide for publish and subscribe, and if not if there is an external data sink that will accept their data; if there is such an external data sink, the Core provides that information to the System User
- Receiving data intended for publish-subscribe from System Users
- Removing any Personally Identifiable Information (PII) that is attached to a data message coming from a user of an anonymous DSRC certificate
- Creating and maintaining System User data subscriptions
- Distributing data according to System User subscriptions

This view also addresses identification of illogical behavior or potential misbehavior by examining some characteristics of the data received as well as the behavior of subscribers. It includes the functionality necessary to instruct data providers how to communicate with external data sinks that may want their data in the case that the Core does not distribute their data. Finally, it includes geo-cast functions to distribute the information necessary for System Users to distribute their information over a geographically-defined area without having knowledge of the specific System Users in that area.

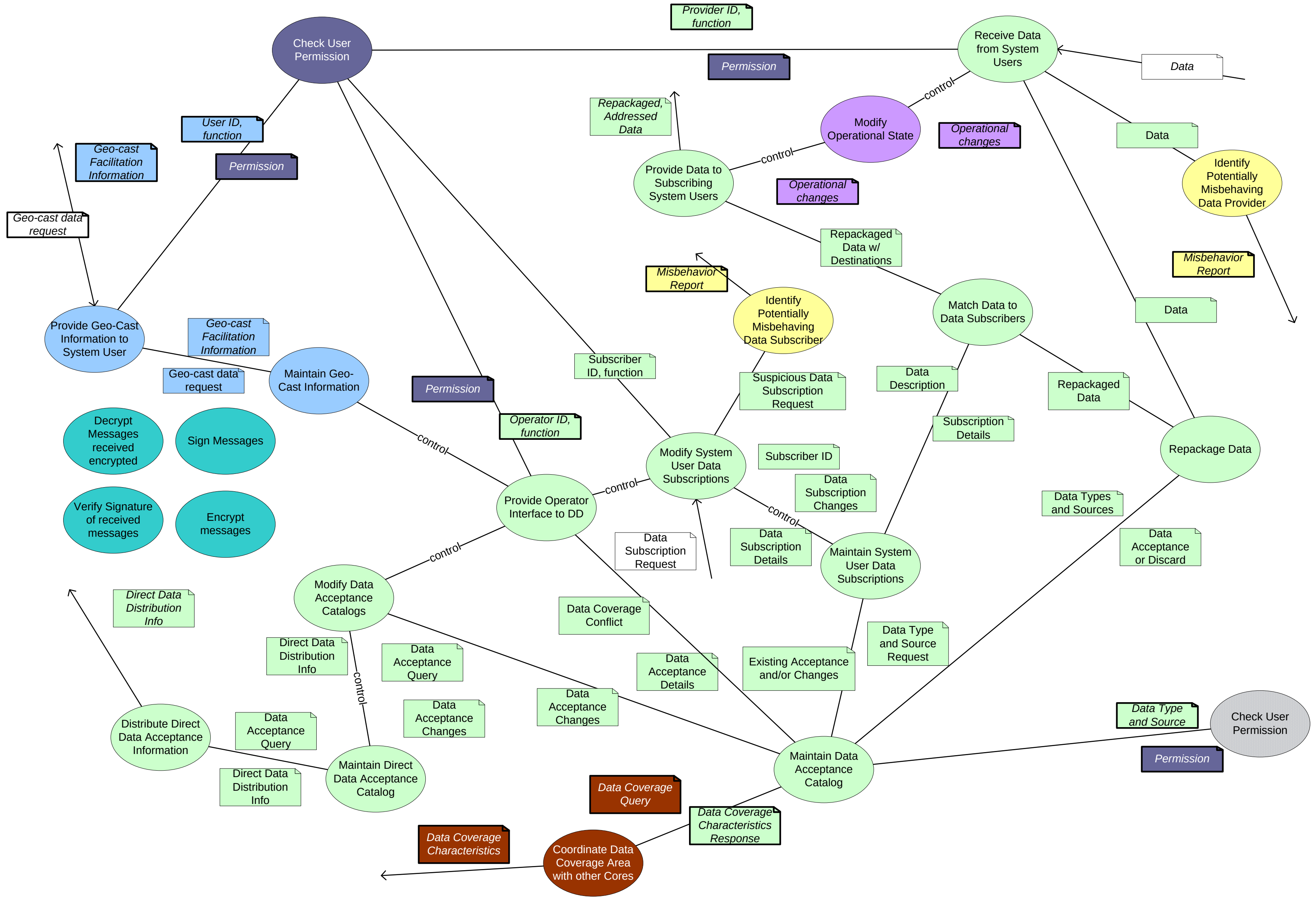

Provider ID, function
Check User Permission
Receive Data from System Users
Permission
Data
control
Repackaged, Addressed Data
Modify Operational State
Operational changes
User ID, function
Geo-cast Facilitation Information
Permission
Data
control
Provide Data to Subscribing System Users
Identify Potentially Misbehaving Data Provider
Operational changes
Geo-cast data request
Repackaged Data w/ Destinations
Misbehavior Report
Misbehavior Report
Identify Potentially Misbehaving Data Subscriber
Match Data to Data Subscribers
Data
Provide Geo-Cast Information to System User
Geo-cast Facilitation Information
Geo-cast data request
Maintain Geo-Cast Information
Subscriber ID, function
Permission
Suspicious Data Subscription Request
Data Description
Repackaged Data
Operator ID, function
control
Subscription Details
Decrypt Messages received encrypted
Sign Messages
Modify System User Data Subscriptions
Subscriber ID
Repackage Data
Provide Operator Interface to DD
control
Data Subscription Changes
Verify Signature of received messages
Encrypt messages
control
Data Types and Sources
Data Subscription Request
Data Subscription Details
Maintain System User Data Subscriptions
Data Acceptance or Discard
control
Modify Data Acceptance Catalogs
Direct Data Distribution Info
Data Coverage Conflict
Data Type and Source Request
Direct Data Distribution Info
Data Acceptance Query
Data Acceptance Details
Existing Acceptance and/or Changes
control
Data Acceptance Changes
Data Acceptance Changes
Data Type and Source
Check User Permission
Distribute Direct Data Acceptance Information
Data Acceptance Query
Maintain Direct Data Acceptance Catalog
Maintain Data Acceptance Catalog
Permission
Direct Data Distribution Info
Data Coverage Query
Data Coverage Characteristics Response
Data Coverage Characteristics
Coordinate Data Coverage Area with other Cores

Figure 4-11: Functional View - Data Distribution

4.2.2.4 Configuration Information

The following views must be considered when changing this view:

Enterprise Viewpoint: Enterprise View – Governance
Connectivity Viewpoint: Connectivity View – Core System Function Allocation
Communications Viewpoint: Communications View – Mobile DSRC Device and Core

4.2.2.5 Alternatives Considered

As written, the Repackage Data function parses incoming data and provides only those component pieces with subscriptions to subscribers. This has a number of advantages and disadvantages:

Advantages:

1. Subscribers receive only those data elements they asked for.
2. Communications resource use is lower than if the entirety of data provider messages were forwarded.
3. Messages with mixed data, including some that the provider may wish to restrict access to, could still yield information to subscribers of the non-restricted data. This enables more complex message formats which may simplify publication, particularly by those constrained by the time windows over which they can interact with the Core (i.e., DSRC-equipped Mobile Users).

Disadvantages:

1. Resource intensive for the Core; this could require a significant amount of processing.
2. Parsing each data packet to extract the data elements that each user wants could be a development challenge.

4.2.3 Functional View – System Monitor and Control

4.2.3.1 Introduction

This view addresses most of the day-to-day Core System operations. It includes elements from Core2Core, Network Services, Service Monitor, Time, User Security, and User Permissions; only Data Distribution and Misbehavior Management are not included. While aspects of the six subsystems included here appear in other views, this view depicts the functionality these subsystems provide to the Core.

Misbehavior Management includes detection of Operator misbehavior which is certainly related to this view. However, Operator misbehavior is not addressed here; it is addressed in Functional View – DSRC Certificate Distribution. That view addresses other forms of misbehavior. Several of the functions required to address other forms of misbehavior are the same as those needed to address Operator misbehavior. Keeping all of the misbehavior in the same view makes the diagrams a simpler.

4.2.3.2 Object Definitions and Roles

Special Objects: Four of these five objects interact with various other functions to perform operations on messages sent or received. The last maintains some common information that is used by many other functions. They are not shown interacting on the diagram for the sake of clarity.

- **Decrypt Messages Received Encrypted:** This function accepts an encrypted message and decrypts it using the Core's private key.
- **Verify Signature of Received Messages:** This function verifies that the signature attached to a message is legitimate using the sender's public key.
- **Sign Messages: This function attaches t**he Core System's digital signature to a message by encrypting a hash of the message content with the Core's private key.
- **Encrypt Messages:** This function encrypts a message using the public key of the intended recipient.
- **Maintain List of External CAs**: This function keeps track of other Certificate Authorities and the Core's relationship with them, including whether it accepts or provides misbehavior and certificate activation information to that CA.

Check User Permission: Accepts a System User ID or Operator ID, along with a type of operation that the user is attempting to access, and responds with whether or not the user is permitted that action.

Associated Information Objects:

- **User ID** is a unique representation of a user, sent to the Maintain User Permission function as a query to determine the user's permissions.
- **User Permissions** is the response from the Maintain User Permission function describing the permissions of the user associated with **User ID**.
- **Operator ID, function** is the identity of the operator which is sent to the Check User Permission function to verify whether the operator is permitted a given **function**.
- **Permission** is the response from the Check User Permission function describing whether or not the **User** or **Operator** is permitted this action.

Generic Application Component: The Generic Application Component function is a representative function representing any other Functional Object.

Associated Information Objects:

- **Time Local Form** is time synchronized with the external source in a format usable by Core functions.
- **Operational Changes** describe how the Functional Object needs to alter operations. If in a normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to normal mode.

Get Time from External Available Source: Acquires time from an external Stratum-2 source and provides it in a usable form to the Make Time Available to All Subsystems function.

Associated Information Objects:

- **Time Local Form** is time synchronized with the external source in a format usable by Core functions.
- **Time** is time acquired from the external Stratum-2 source.

Log System State and Performance: Accepts Core performance, monitoring and configuration changes, and logs that information for subsequent analysis performed by external support systems.

Associated Information Objects:

- **Detailed Service Status** is detailed information describing the performance of all Core functions and interfaces, received from the Monitor Core Services Performance function.
- **Physical Security Status** is a summary of the power, environmental and physical security conditions of the Core System, received from the Monitor Core Physical Security function.
- **Health and Safety Status** is the status of the integrity of operating Core software and hardware, received from the Monitor Core Health and Safety function.

Maintain Core Operational Configuration: Allows modification of the Core's configuration. This includes addition of functions and interfaces, additional hardware, and additional instantiations of existing functions. It also allows the Operator to set priorities for Core function and for different System Users and System User types.

Associated Information Objects:

- **Configuration Settings** are the existing configuration data describing Core configuration sent to the Provide Operator Interface function.
- **Configuration Changes** are Operator-provided changes to the Core's configuration.

Maintain Geo-cast Information: Keeps track of the information necessary for System Users to perform geo-casting. This includes locations, addresses, and ranges of devices that accept geo-cast messages.

Associated Information Objects:

- **Geo-cast Info** describes geo-cast supporting devices, including performance and operating characteristics, and the permissions System Users must have to access them.
- **Geo-cast Info Changes** describes changes to the geo-cast info of geo-cast supporting devices; this includes performance and operating characteristics, and the permissions System Users must have to access these devices.

Maintain User Permissions: Allows update and creation of System Users and Operators and how they are allowed to interact with the Core System. This includes functions they may exercise and characteristics of their use such as frequency and source location (e.g., a given user may be permitted as a Center User but not as a Mobile User). This function maintains the certificate-managed application permissions that a user is permitted.

Associated Information Objects:

- **User ID** is a unique representation of a user received from the Check User Permission or Provide Operator Interface functions as a query to determine the user's permissions and/or characteristics
- **User Permissions** is the response to the Check User Permission function describing the permissions of the user associated with **User ID**.
- **User Info** describes the characteristics of a given **User** and is provided to the Provide Operator Interface function.
- **User Info Changes** are changes to a User's permissions received from the Provide Operator Interface function.

Make Time Available to All Subsystems: Provides the syncs the time acquired externally with the time available to all Core System components.

Associated Information Objects:

- **Time Local Form** is time synchronized with the external source in a format usable by Core functions.

Modify Operational State: This is a controlling function that instructs various functions to change the way they operate. For example, in restricted mode, Receive Data from System Users will be instructed to stop receiving data except from specified users, and Provide Data to Subscribing users similarly restricted to provide data only to specified users (see the Data Distribution functional view for more information).

Associated Information Objects:

- **Operational Changes** describe how the Functional Object needs to alter operations. If in a normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to normal mode.

Monitor Core Health and Safety: Monitors the integrity of Core System components. If a component fails an integrity check or is determined to be out of date or otherwise compromised, it reports that information to other functions including Provide Operator Interface.

Associated Information Objects:

- **Health and Safety Status** is the status of the integrity of operating Core software and hardware.

Monitor Core Physical Security: Monitors the status of the Core System's physical security. This includes physical locking mechanisms to control access to Core hardware, the environmental conditions in which the Core hardware resides including temperature and humidity, and the Core's power systems.

Associated Information Objects:

- **Physical Security Status** is a summary of the power, environmental and physical security conditions of the Core System sent to other functions that may act on this information, including Provide Operator Interface.

Monitor Core Services Performance: Monitors the performance of all Core System functions and interfaces. This includes measures of throughput, buffer levels, and resource usage. This information is provided to the Log System State and Performance and Provide Operator Interface functions; summary information describing the state of Core services is provided to the Provide Service Status to System Users and Provide Service Status to other Cores functions.

Associated Information Objects:

- **Detailed Service Status** is detailed information describing the performance of all Core functions and interfaces provided to the Log System State and Performance and Provide Operator Interface functions.
- **Service Status** is summary information describing the state of Core services provided to the Provide Service Status to System Users and Provide Service Status to other Cores functions.

Provide Operator Interface: Provides a user interface to the Operator. This function interacts with many other monitoring, configuration and settings functions to enable the Operator to manage the Core.

Associated Information Objects:

- **Geo-cast Info** describes geo-cast supporting devices, including performance and operating characteristics, and the permissions System Users must have to access them.
- **Geo-cast Info Changes** describes changes to the geo-cast info of geo-cast supporting devices; this includes performance and operating characteristics, and the permissions System Users must have to access these devices.
- **User Info** describes the characteristics of a given **User** received from the Maintain User Permissions function.
- **User Info Changes** are changes to a User's characteristics and/or permissions sent to the Maintain User Permissions function.
- **User ID** is a unique representation of a user sent to the Maintain User Permissions function as a query to determine the user's characteristics and permissions.
- **Other Core Status** is received from the Monitor Status of Other Cores function and describes the status of another Core's services including operating state and mode.
- **Environmental Response** is a description of the recommended or automatically imposed action this function suggests or takes in response to physical or system health issues; received from the Take Action in Response to Environmental Issue function.
- **Physical Security Status** is a summary of the power, environmental, and physical security conditions of the Core System; received from the Monitor Core Physical Security function.
- **Health and Safety Status** is the status of the integrity of operating Core software and hardware received from the Monitor Core Health and Safety function.
- **Detailed Service Status** is detailed information describing the performance of all Core functions and interfaces, received from the Monitor Core Services Performance function.
- **Configuration Settings** are the existing configuration data describing Core configuration sent received from the Maintain Core Operational Configuration function.

- **Configuration Changes** are Operator-provided changes to the Core's configuration, sent to the Maintain Core Operational Configuration function.

Provide Service Status to other Cores: Provides the status of Core services on a subsystem basis to other Cores.

Associated Information Objects:

- **Core Service Status Query** is a request for Core service status information from another Core.
- **Service Status for Cores** is the status of the Core's subsystems, including state, mode, and performance data, sent to other Cores.

Provide Service Status to System Users: Provides the status of Core services on a subsystem basis to System Users.

Associated Information Objects:

- **Core Service Status Query** is a request for Core service status information from a System User.
- **Service Status for Cores** is the status of the Core's subsystems, including state and mode, sent to System Users.

Monitor Status of other Cores: Queries other Cores to determine the status of their services.

Associated Information Objects:

- **Core Service Status Query** is query sent to another Core to determine the status of that Core's services.
- **Other Core Status** is received from another Core; it describes the status of that Core's services including operating state and mode. This message is also passed to the Provide Operator Interface function for presentation to the Operator.

Take Action in Response to Physical Environmental Issue: Reacts to changes in the Core's physical environment, including power, temperature, humidity or physical intrusions noted by Monitor Core Health and Safety and Monitor Core Physical Security. Depending on the issue and Core configuration, it may either recommend action to the Operator or automatically initiate a change of state.

Associated Information Objects:

- **Environmental Response** is a description of the recommended or automatically imposed action this function suggests or takes in response to physical or system health issues.
- **Physical Security Status** is a summary of the power, environmental, and physical security conditions of the Core System; received from the Monitor Core Physical Security function.
- **Health and Safety Status** is the status of the integrity of operating Core software and hardware, received from the Monitor Core Health and Safety function.

4.2.3.3 View Description

This view identifies day-to-day housekeeping functions that enable the Operator to manage the Core. It also provides operational status and performance information to qualified users.

Service Monitor functions provide the user interface for the Operator, monitor the Core System's physical state including physical security and environmental conditions, monitor the performance of Core functions, and monitor the integrity of Core components and functions. All of this information is stored and made available to the Operator. Some of this information is shared with System Users and other Cores.

Service Monitor’s Operator interface also provides means for the Operator to configure Core System functions and components, create and modify users, and monitor and modify the operational state and mode of individual functions.

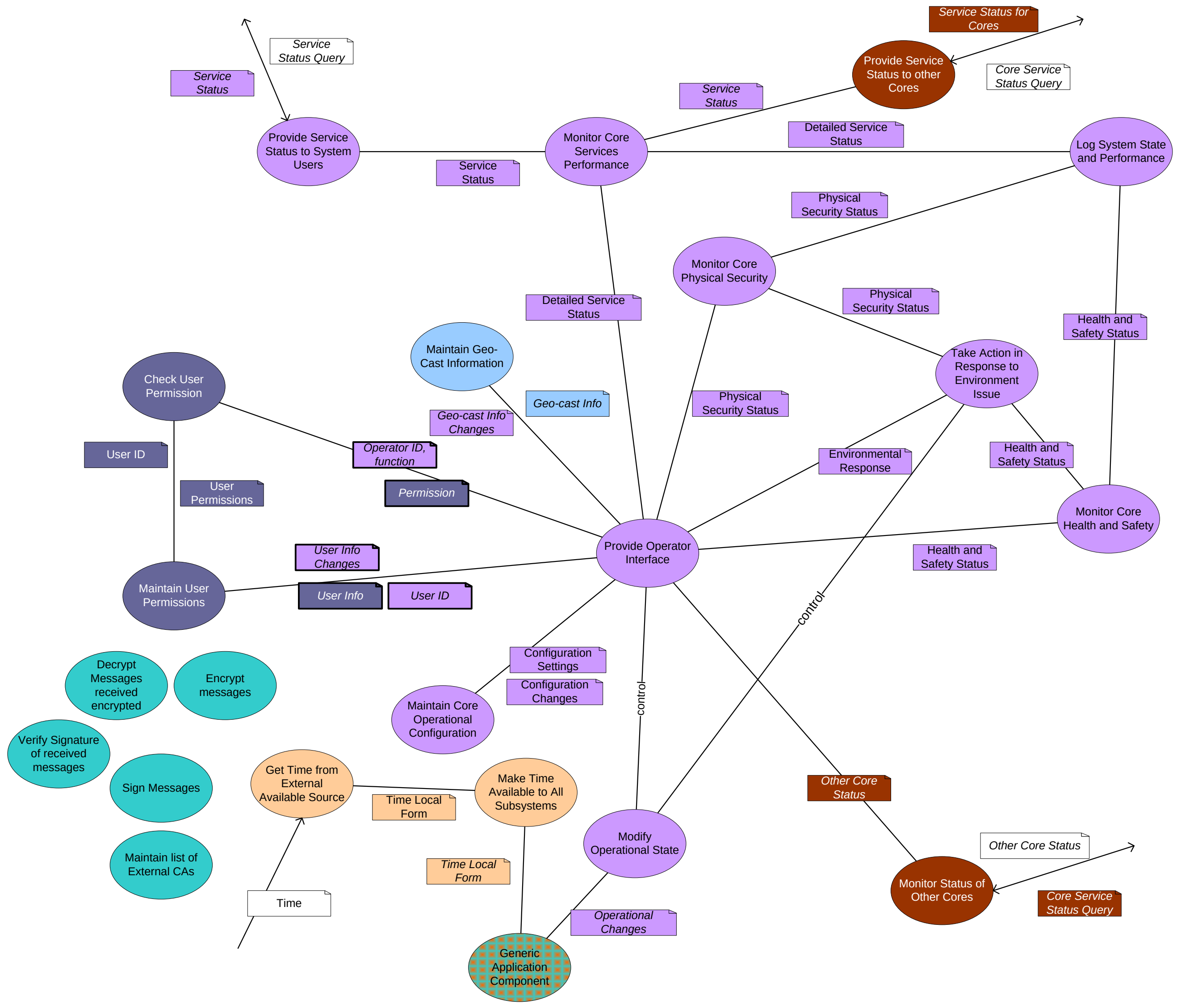

Service Status for Cores
Service Status Query
Service Status
Provide Service Status to other Cores
Core Service Status Query
Service Status
Detailed Service Status
Provide Service Status to System Users
Service Status
Monitor Core Services Performance
Log System State and Performance
Physical Security Status
Monitor Core Physical Security
Physical Security Status
Detailed Service Status
Health and Safety Status
Maintain Geo-Cast Information
Check User Permission
Take Action in Response to Environment Issue
Geo-cast Info
Geo-cast Info Changes
Physical Security Status
User ID
Operator ID, function
Health and Safety Status
Environmental Response
User Permissions
Permission
Monitor Core Health and Safety
User Info Changes
Provide Operator Interface
Health and Safety Status
Maintain User Permissions
User Info
User ID
control
Configuration Settings
Decrypt Messages received encrypted
Encrypt messages
Configuration Changes
Maintain Core Operational Configuration
control
Verify Signature of received messages
Get Time from External Available Source
Make Time Available to All Subsystems
Sign Messages
Time Local Form
Other Core Status
Modify Operational State
Other Core Status
Maintain list of External CAs
Time Local Form
Time
Monitor Status of Other Cores
Core Service Status Query
Operational Changes
Generic Application Component

Figure 4-12: Functional View - Service Monitor and Control

4.2.3.4 Configuration Information

The following views must be considered when changing this view:

None.

4.2.3.5 Alternatives Considered

None.

4.2.4 Functional View – DSRC Certificate Distribution

4.2.4.1 Introduction

Creation and distribution of certificates, including certificate revocation and the distribution of certificate related roles (i.e., registration versus certificate distribution) is one of the primary roles for the Core. How many Cores need to be involved in distributing certificates? Ensuring trust is a need that all Cores must meet, but if one Core provides certificates and all other Cores refer System Users to that Core, then the need is met but only one Core need perform the intricacies of certificate management.

The primary view documented here includes certificate distribution by the Core, but alternative views are presented where the Core does not perform this function. These two concepts could exist together, so that one Core is the provider of DSRC certificates for all DSRC users, but other Cores work with that Core to facilitate certificate distribution.

A certificate includes a public key corresponding to a private key held by the certificate's owner; the certificate has been signed by a CA which all parties trust. The public-key/private-key pair are generated by the CA and distributed to System Users that the CA has verified the identity of and confirmed is entitled to have a public key certificate. This verification task can be performed by a Registration Authority (RA) to offload the CA.

This view illustrates the Core serving as both certificate authority and registration authority. Alternative views show what functionality is removed or changed when registration and/or certificate distribution functions are performed outside the Core.

4.2.4.2 Object Definitions and Roles

Special Objects: Four of these five objects interact with various other functions to perform operations on messages sent or received. The last maintains some common information that is used by many other functions. They are not shown interacting on the diagram for the sake of clarity.

- **Decrypt Messages Received Encrypted:** This function accepts an encrypted message and decrypts it using the Core's private key. In case of a decryption error it provides a description of the error along with a copy of the encrypted message to Receive Internal Misbehavior Reports.
- **Verify Signature of Received Messages:** This function verifies that the signature attached to a message is legitimate using the sender's public key. In case of a verification error it provides a description of the error along with a copy of the message to Receive Internal Misbehavior Reports.
- **Sign Messages:** This function attaches the Core System's digital signature to a message by encrypting a hash of the message content with the Core's private key.
- **Encrypt Messages:** This function encrypts a message using the public key of the intended recipient.
- **Maintain List of External CAs**: This function keeps track of other Certificate Authorities and the Core's relationship with them, including whether it accepts or provides misbehavior and certificate activation information to that CA.

Check User Permission: This function accepts a System User ID or Operator ID, along with a type of operation that the user is attempting to access, and responds with whether or not the user is permitted that action. With regard to certificate distribution User Permissions maintains a listing of what each user is permitted to do; consequently it provides the information necessary to establish the permissions section of a DSRC certificate.

Associated Information Objects:

- **User ID, function** is the identity of the System User querying for geographic information distribution sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Subscribing ID, function** is the identity of the System User data subscriber sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Provider ID, function** is the identity or type of the System User data provider (which could be anonymous) sent to the Check User Permission function to verify whether the System User is permitted a given **function**.
- **Operator ID, function** is the identity of the operator sent to the Check User Permission function to verify whether the operator is permitted a given **function**.
- **Permission** is the response from the Check User Permission function describing whether or not the **User** is permitted this action.
- **Certificate Owner ID** is received from Provide DSRC Identity Certificates as a query to determine whether the System User is allowed to request a DSRC identity certificate.
- **Cert Permission** is the response sent to Provide DSRC Identity Certificates indicating whether the System User is allowed to request a DSRC identity certificate and also the types of permissions that must be attached to the identity certificate.
- **System User ID, function** is a request received from Identify Misbehaving System Users to determine whether the System User is permitted to carry out **function**.
- **SUID, Function Permission** is a response sent to Identify Misbehaving System Users indicating whether or not the System User is permitted to use the **function**.

Coordinate Certificate Distribution with Other Cores: Facilitates sharing certificate distribution tasks with other Cores. This function distributes the IDs of the anonymous and identity certificates that the Core has distributed with other Cores, and receives the same from those other Cores.

Associated Information Objects:

- **Activated DSRC Anonymous Certificate IDs** are the identifiers of DSRC anonymous certificates that the Core has distributed to System Users. This message is received from the Provide DSRC Anonymous Certificates function, and also passed on from other Cores to the Maintain DSRC Anonymous Certificates function.
- **Activated DSRC Identity Certificate IDs** are the identifiers of DSRC identity certificates that the Core has distributed to System Users. This message is received from the Provide DSRC Identity Certificates function, and also passed on from other Cores to the Maintain DSRC Identity Certificates function.

Distribute DSRC CRL: Distributes the DSRC Certificate Revocation List to System Users.

Associated Information Objects:

- **CRL** is the list of active certificates that are no longer valid.

Exchange CRLs with other Cores: This function allows the Core to exchange Certificate Revocation Lists (CRLs) with other Cores. This allows each Core to maintain a complete CRL, so that System Users need receive a CRL from only one source.

Associated Information Objects:

- **Complete CRL** is a message containing the IDs of all certificates the sending Core knows to be invalid. This is sent by the Core or received from another Core.
- **CRL Deltas** is a message describing the changes to the CRL since the last CRL Delta or Complete CRL was sent to the Core that is receiving this message.

Exchange Misbehavior Reports with other Cores: Establishing a pattern of behavior sufficient to withdraw System User permissions may not occur unless Cores share reports, allowing them to draw conclusions based on multiple instances of misbehavior. This function allows Cores to share misbehavior reports. If operating in a restricted mode, this function will lower its priority, and possibly cease receiving misbehavior reports based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Misbehavior Report** is a message containing a certificate ID, time, and type of misbehavior identified.
- **Operational Changes** describe how this function needs to alter operations. If in normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions including possibly removing all restrictions and returning to normal mode.

Identify Misbehaving Operators: Monitors Operator actions and determines whether those actions constitute misbehavior.

Associated Information Objects:

- **Operator ID, misbehavior** is a message sent to the Revoke Operator Permissions function, sent when the Identify Misbehaving Operators function identifies misbehavior sufficient to justify revoking Operator permissions. The message includes the Operator's ID and an indication of the types of permissions to revoke.
- **Operator ID, function** is the identity of the operator sent to the Check User Permission function to verify whether the operator is permitted a given **function**.
- **Permission** is the response from the Check User Permission function describing whether or not the Operator is permitted this action.
- **Misbehaving Operator Alert** is sent to the Provide Operator Interface to Misbehavior Management (MM) function to inform other Operators that an Operator may be engaged in misbehavior.

Identify Misbehaving System Users: Examines System User misbehavior reports and identifies when users are malfunctioning or appearing to act maliciously.

Associated Information Objects:

- **Misbehavior Report Request** requests misbehavior reports from the Manage Misbehavior Reports function according to a set of criteria depending on the type of analysis to be done. The message may indicate reports that happened at a certain time interval, spatial restriction, System User type or even specific System User.

- **Misbehavior Reports for Analysis** sent from the Manage Misbehavior Reports function contains the misbehavior reports requested by the Misbehavior Report Request.
- **Misbehaving User Alert** is sent to the Provide Operator Interface to MM function to inform Operators that a System User may be engaged in misbehavior. The message indicates the type of misbehavior and type and/or identity of the System User as applicable.
- **System User ID, function** is a request sent to Check User Permissions to determine whether the System User is permitted to carry out **function**.
- **SUID, Function Permission** is a response from Check User Permissions indicating whether or not the System User is permitted to use the **function**.

Maintain DSRC Anonymous Certificates: Maintains the inventory of DSRC anonymous certificates distributed by the Core including the key pairs associated with these certificates. It also maintains a list of the identifiers of identity certificates issued by other Cores. It provides these certificates upon valid request.

Associated Information Objects:

- **Anonymous Cert Request** is a request for a new groups of anonymous certificates sent from the Provide DSRC Anonymous Certificates function on behalf of a System User.
- **DSRC Anonymous Certs** is a message containing one or more new anonymous DSRC certificates for the System User.
- **Activated DSRC Anon Cert IDs** contains the IDs of the DSRC anonymous certificates issued to a System User by other Cores; provided by the Coordinate Certificate Distribution with Other Cores function.

Maintain DSRC Identity Certificates: Maintains the inventory of DSRC identity certificates distributed by the Core including the key pairs associated with these certificates. It also maintains a list of the identifiers of identity certificates issued by other Cores. It provides these certificates upon valid request.

Associated Information Objects:

- **DSRC Identity Cert Request** is a request for a new identity certificate sent from the Provide DSRC Identity Certificates function on behalf of a System User.
- **DSRC Identity Certs** is a message containing one or more new identity certificates for the System User.
- **Activated DSRC Identity Cert IDs** contains the IDs of the DSRC identity certificates issued to a System User by other Cores; provided by the Coordinate Certificate Distribution with Other Cores function.

Manage DSRC CRL: Maintains the Certificate Revocation List for all DSRC certificates, including both anonymous and identity certificates. It adds entries to the CRL based on inputs from Manually Confirm/Identify Misbehaving Users. It responds to queries about associations between System User IDs and pseudo-IDs with entries in the CRL.

Associated Information Objects:

- **Anon Cert Pseudo-ID** is received from Provide DSRC Anonymous Certificates; it includes a pseudo-ID associated with an anonymous certificate.
- **Anon CRL Associations** a response to the Provide DSRC Anonymous Certificates function that indicates the certificate IDs that are on the CRL and are associated with the pseudo-ID previously provided.
- **CRL** is the complete certificate revocation list.

- **CRL Deltas** is a message containing changes to the CRL since the last time it was published.
- **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed and is received from the Manually Confirm/Identify Misbehaving Users function.
- **System User ID** is received from Provide DSRC Identity Certificates and includes the identity of a System User.
- **System User CRL Associations** is a response to the Provide DSRC Identity Certificates function that indicates the certificate IDs that are on the CRL and are associated with the System User ID previously provided.

Manage Misbehavior Reports: Maintains the misbehavior reports received by the Core, including those submitted by System Users, and those from other Cores and those generated by internal Core monitoring processes. It accepts queries on those reports and provides the reports matching that query to requesting functions.

Associated Information Objects:

- **System User Misbehavior Reports** is a report of misbehavior from the Receive System User Misbehavior Reports function. It includes an identification of the reporting System User, ID or pseudo-ID of the misbehaving user, and characterization of the type of misbehavior.
- **Internal Misbehavior Reports** is a report of misbehavior from the Receive Internal Misbehavior Reports function. It includes an identification of the reporting function, ID or pseudo-ID of the misbehaving user, and characterization of the type of misbehavior.
- **Misbehavior Report Request** is a request for misbehavior reports from the Identify Misbehaving System Users function. It includes a set of criteria for report selection including time interval, spatial restriction, System User type or even specific System User.
- **Misbehavior Reports for Analysis** sent to the Identify Misbehaving System Users function contains the misbehavior reports requested by that function.

Manually Confirm/Identify Misbehaving Users: Accepts input from the Provide Operation Interface to MM function, and thus from the Operator of a misbehaving System User and his misbehavior. This allows the Operator to manually identify misbehaving users.

Associated Information Objects:

- **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed and is provided to the Manage DSRC CRL and Revoke User Permissions functions for further action.

Modify Operational State: is a controlling function that instructs various components to change the way they operate. In restricted mode, exchange of misbehavior reports and distribution of certificates will be constrained.

Associated Information Objects:

- **Operational Changes** describe how the certificate distribution, misbehavior report exchange and misbehavior acquisition functions need to alter operations. If in a normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to normal mode.

Provide DSRC Anonymous Certificates: Accepts requests for DSRC anonymous certificates from System Users and services valid requests by obtaining anonymous Certificates from the Maintain DSRC Anonymous Certificates function and passing them along to the System User. It verifies the certificate request by examining the provided certificate and verifying its format and content. If operating in a restricted mode this function will prioritize certificate requests based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Anon Cert Update Request** is a request for a new anonymous certificate from a System User that had a previous anonymous certificate. It includes the System User's last used DSRC anonymous certificate.
- **Anon Cert New Request** is a request for a new anonymous certificate from a System User. It includes the System User's identity.
- **Activated DSRC Anon Cert IDs** are a list of the IDs of issued certificates provided to the Coordinate Certificate Distribution with Other Cores function.
- **DSRC Anon Certs** is a message containing one or more new anonymous certificates for the System User.
- **Anon Cert Pseudo-ID** is a query to the Maintain DSRC CRL function including the last used Pseudo-ID of the System User requesting certificates.
- **Anon CRL Associations** is the response to the Pseudo-ID query identifying revoked certificates associated with the Pseudo-ID.
- **Activated DSRC Anonymous Certificate IDs** are the identifiers of DSRC anonymous certificates that the Core has distributed to System Users. This message is provided to the Coordinate Certificate Distribution with Other Cores function.
- **Operational Changes** describe how this function needs to alter operations. If in normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to normal mode.

Provide DSRC Identity Certificates: Accepts requests for DSRC identity certificates from System Users, and services valid requests by obtaining identity certificates from the Maintain DSRC Identity Certificates function and passes them along to the System User. It verifies the certificate request by examining the provided certificate and verifying its format and content, and then querying Check User Permissions to verify that the owner of this certificate is allowed to request new certificates. If operating in a restricted mode this function will prioritize certificate requests based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Certificate Request** is a request for a new identity certificate from a System User.
- **DSRC Identity Certs** is a message a new identity certificates for the System User.
- **DSRC Identity Cert Request** is a message to the Maintain DSRC Identity Certificates function requesting a new identity certificate on behalf of the System User.
- **Activated DSRC Identity Cert IDs** contains the IDs of the DSRC identity certificates issued to a System User, provided to the Coordinate Certificate Distribution with Other Cores function.

- **Certificate Owner ID** is sent to Check User Permissions as a query to determine whether the System User is allowed to request a DSRC identity certificate.
- **Cert Permission** is the response from Check User Permissions indicating whether the System User is allowed to request a DSRC identity certificate, and also the types of permissions that must be attached to the identity certificates.
- **System User ID** is a query sent to Manage CRL that includes the identity of a System User, asking for any certificate IDs associated with this user that are on the CRL.
- **System User CRL Associations** is a response from the Manage CRL function that indicates the certificate IDs that are on the CRL and are associated with the System User ID previously provided.
- **Operational Changes** describe how this function needs to alter operations. If in normal mode, this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to normal mode.

Provide Operator Interface to MM: Provides an interface to the Operator allowing him access to Misbehavior Management functions.

Associated Information Objects:

- **Misbehaving Operator Alert** is received from the Identify Misbehaving Operator function to inform other Operators that an Operator may be engaged in misbehavior.

.

Receive Internal Misbehavior Reports: Accepts misbehavior reports from other functions within the Core. It validates reports and passes them to the Manage Misbehavior Reports function.

Associated Information Objects:

- **Internal Misbehavior Reports** is a report of misbehavior received from the Check User Permissions function and sent to the Manage Misbehavior Reports function. It includes an identification of the reporting function, ID or pseudo-ID of the misbehaving user, and characterization of the type of misbehavior.
- **Decryption Error Message** describes a decryption failure and includes a copy of the encrypted message.
- **Signature Error Message** describes a signature verification failure, and includes a copy of the message that failed signature verification.

Receive System User Misbehavior Reports: This function accepts a misbehavior reports from System Users describing alleged misbehavior of other System Users. It validates reports and passes them to the Manage Misbehavior Reports function. If operating in a restricted mode, this function will lower its priority, and possibly cease receiving misbehavior reports based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **System User Misbehavior Report** is a report of misbehavior received from System Users and subsequently sent to the Manage Misbehavior Reports function. It includes an identification of the reporting System User, ID or pseudo-ID of the misbehaving user, and characterization of the type of misbehavior.
- **Operational Changes** describe how this function needs to alter operations. If in normal mode, this will include restrictions on operations. If already restricted, this will be a

change to restrictions, including possibly removing all restrictions and returning to normal mode.

Revoke Operator Permissions: Accepts an Operator ID and an identification of misbehavior the Operator is engaged in and subsequently revokes his permissions through the Check User Permissions function.

Associated Information Objects:

- **Operator ID, misbehavior** is a message received from the Identify Misbehaving Operators function. The message includes the Operator's ID and an indication of the types of permissions to revoke.
- **Operator ID, function** is a request sent to Check User Permissions to revoke the Operator's permissions to carry out the indicated function.
- **Permissions** is a response from Check User Permissions identifying the Operator's new permissions.

Revoke User Permissions: Accepts a System User ID and an identification of misbehavior the System User is engaged in, and subsequently revokes his permissions through the Check User Permissions function.

Associated Information Objects:

- **System User ID, misbehavior** is a message received from the Manually Confirm/Identify Misbehaving Users function. The message includes the System User ID, and an indication of the types of permissions to revoke.
- **System User ID, function** is a request sent to Check User Permissions to revoke the System User's permissions to carry out the indicated function.
- **Permissions** is a response from Check User Permissions identifying the System User's new permissions.

4.2.4.3 View Description

This view includes the functions necessary act as a DSRC Certificate Authority (i.e., distribution and management of DSRC certificates) and detection of misbehavior. Within those broad high level functions:

- Distribution of DSRC certificates includes the distribution of both anonymous and identity certificates including the generation of the public and private keys necessary to encrypt and decrypt messages and the maintenance and distribution of Certificate Revocation Lists (CRLs). It also includes the exchange of certificate distribution coordination information between Cores and the ability to restrict certificate distribution when the Core changes operational state.

- Detection of misbehavior includes the detection of misbehavior by Field, Center, and Mobile Users and Operators, using data reported by non-anonymous System Users, and detection mechanisms inside the Core. Misbehavior detection can be done independently of certificate distribution, but acting on detection of misbehavior includes modification of the CRL, which can only effectively be done by the entity responsible for certificate distribution. Therefore functions included in misbehavior detection are restricted in the alternative considering CA functionality outside the Core.

- Misbehaving operators will not have DSRC certificates and in fact may be associated with X.509 certificates. However detection of misbehaving operators is shown on this view because most of the other misbehavior-related functions are shown here.

Both certificate distribution and misbehavior management require coordination between Cores to ensure that CRLs are consistent, that no two Cores issue the same certificates, and to improve misbehavior detection by sharing misbehavior reports between Cores. If more than one Core serves as DSRC Certificate Authority, only one Core will serve as Root. Other Cores will serve as sub-CA.

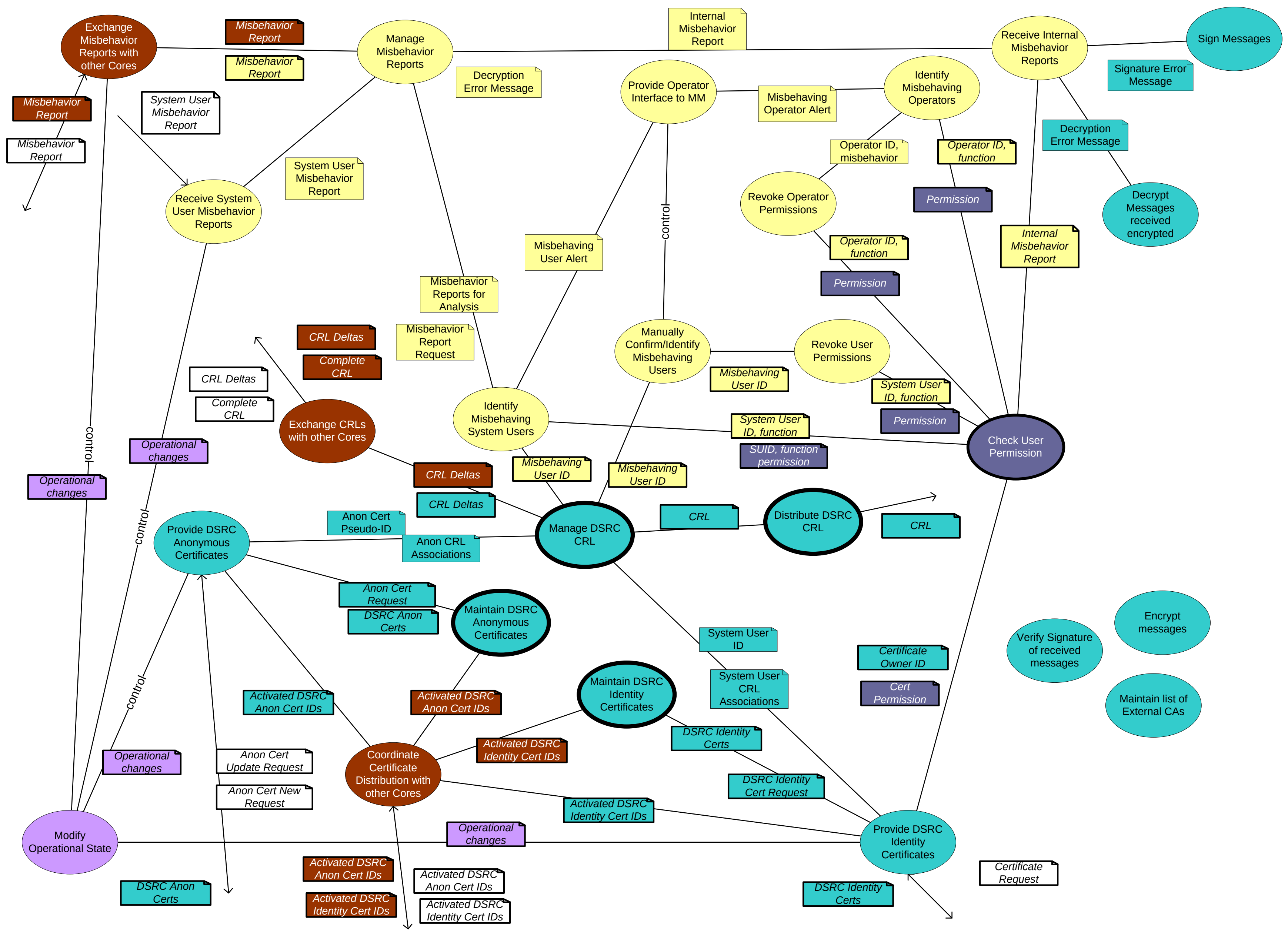

Exchange Misbehavior Reports with other Cores
Misbehavior Report
Misbehavior Report
Manage Misbehavior Reports
Internal Misbehavior Report
Receive Internal Misbehavior Reports
Sign Messages
Signature Error Message
Misbehavior Report
System User Misbehavior Report
Misbehavior Report
Decryption Error Message
Provide Operator Interface to MM
Misbehaving Operator Alert
Identify Misbehaving Operators
Decryption Error Message
Operator ID, misbehavior
Operator ID, function
Receive System User Misbehavior Reports
System User Misbehavior Report
Revoke Operator Permissions
Permission
Decrypt Messages received encrypted
Misbehaving User Alert
control
Operator ID, function
Internal Misbehavior Report
Permission
Misbehavior Reports for Analysis
CRL Deltas
Misbehavior Report Request
Complete CRL
Manually Confirm/Identify Misbehaving Users
Revoke User Permissions
CRL Deltas
Misbehaving User ID
System User ID, function
Complete CRL
Identify Misbehaving System Users
Exchange CRLs with other Cores
System User ID, function
Permission
Check User Permission
control
Operational changes
SUID, function permission
Operational changes
Misbehaving User ID
Misbehaving User ID
CRL Deltas
CRL Deltas
control
Anon Cert Pseudo-ID
CRL
Distribute DSRC CRL
Provide DSRC Anonymous Certificates
Manage DSRC CRL
Anon CRL Associations
CRL
Anon Cert Request
DSRC Anon Certs
Maintain DSRC Anonymous Certificates
Encrypt messages
Verify Signature of received messages
System User ID
Certificate Owner ID
control
Maintain DSRC Identity Certificates
System User CRL Associations
Activated DSRC Anon Cert IDs
Activated DSRC Anon Cert IDs
Cert Permission
Maintain list of External CAs
DSRC Identity Certs
Activated DSRC Identity Cert IDs
Operational changes
Anon Cert Update Request
Coordinate Certificate Distribution with other Cores
DSRC Identity Cert Request
Anon Cert New Request
Activated DSRC Identity Cert IDs
Modify Operational State
Operational changes
Provide DSRC Identity Certificates
DSRC Anon Certs
Activated DSRC Anon Cert IDs
Activated DSRC Anon Cert IDs
Certificate Request
DSRC Identity Certs
Activated DSRC Identity Cert IDs
Activated DSRC Identity Cert IDs

Figure 4-13: Functional View - DSRC Certificate Management

4.2.4.4 Configuration Information

The following views must be considered when changing this view:

Enterprise Viewpoint: Enterprise View – DSRC Security Credentials Configuration. In particular, the **Anon Cert New Request** object used by **Provide DSRC Anonymous Certificates** may be affected. If the Core acts as RA, this object is required. If the Core does not act as RA, this object is not required.
Enterprise Viewpoint: Enterprise View – Governance
Functional Viewpoint: Functional View – X.509 Certificate Management

4.2.4.5 Alternatives Considered

1) Certificates may be pre-assigned to DSRC users, and stored encrypted in an inactive state. Certificates are activated upon request and response from the Core. In this case, certificates do not need to be transmitted, but activation codes do. The information objects DSRC Anon Certs and DSRC Identity Certs become activation codes instead. There are no other changes to the diagram. This approach is compatible with the approach documented above; i.e. some System Users could have pre-loaded certificates while others need the Core to provide them the whole certificate.
2) Registration may be performed outside the Core that performs certification. In other words, one Core provides CA, and another core functions as RA. For the CA Core the functions Provide DSRC Identity Certificates and Provide DSRC Anonymous Certificates are modified, removing the verification checking from those functions. The link between Provide DSRC Identity Certificates and Check User Permissions would disappear and be replaced by a similar link from Maintain DSRC Identity Certificates, used to determine the permissions to attach to identity certificates. There are no other changes to the CA-Core's diagram

   This approach has the advantage of splitting knowledge of a System User's true identity between two entities (CA and RA), helping to ensure that the System User's privacy would be difficult to compromise. This approach is compatible with the approach documented above, as long as at least one Core serves as CA.

   Figure 4-14 illustrates the Core that performs RA functions only. Several functions from the basic architecture are deleted. Three additional objects are defined:

   Provide Misbehaving System User Info to Other Cores: Provides the results of Misbehavior Management to other Cores. Misbehavior detection must still be handled by the Core because the RA is familiar with the user's identity and roles, not the CA. This function provides the CA-core with the ID of misbehaving System Users and provides information necessary to revoke or modify their certificate.

   Associated Information Objects:

   - **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed and is provided to other Cores that perform CA functions for further action.

Request Certificate: This function accepts a certificate request with associated permissions from the Verify DSRC User Identity function and passes that request to a Core with CA functionality. Certificates received from that Core are provided to System User.

Associated Information Objects:

- **Certificate Request** is a request for a new certificate from a System User. This can be a request for an anonymous certificate or an identity certificate.
- **Certificate Request, Permissions** is a message to another Core with CA functionality requesting certificates with the attached permissions for the System User. This can be a request for an anonymous certificate or an identity certificate.
- **DSRC Anon Certs** is a message containing one or more new anonymous certificates for the System User. This message originates from a Core with CA functionality, is sent to the Request Certificate function, then to the Verify DSRC User Identity function, and is then provided to the System User.
- **DSRC Identity Certs** is a message a new identity certificates for the System User. This message originates from a Core with CA functionality, is sent to the Request Certificate function, then to the Verify DSRC User Identity function, and is then provided to the System User.

Verify DSRC User Identity: This function accepts requests for DSRC certificates from System Users, verifies the certificate request by examining the provided certificate and verifying its format and content, and then querying Check User Permissions to verify that the owner of this certificate is allowed to request new certificates. It then passes a message to the Request Certificate function, requesting the certificate for the System User in question.

Associated Information Objects:

- **Certificate Request** is a request for a new identity certificate from a System User. This can be a request for an anonymous certificate or an identity certificate.
- **Cert Permission** is the response from Check User Permissions indicating whether the System User is allowed to request a DSRC identity certificate, and also the types of permissions that must be attached to the identity certificates.
- **Certificate Request, Permissions** is a message sent to the Request Certificate function, containing a certificate request and the permissions the System User is entitled to.
- **Certificate Owner ID** is sent to Check User Permissions as a query to determine whether the System User is allowed to request a DSRC identity certificate.

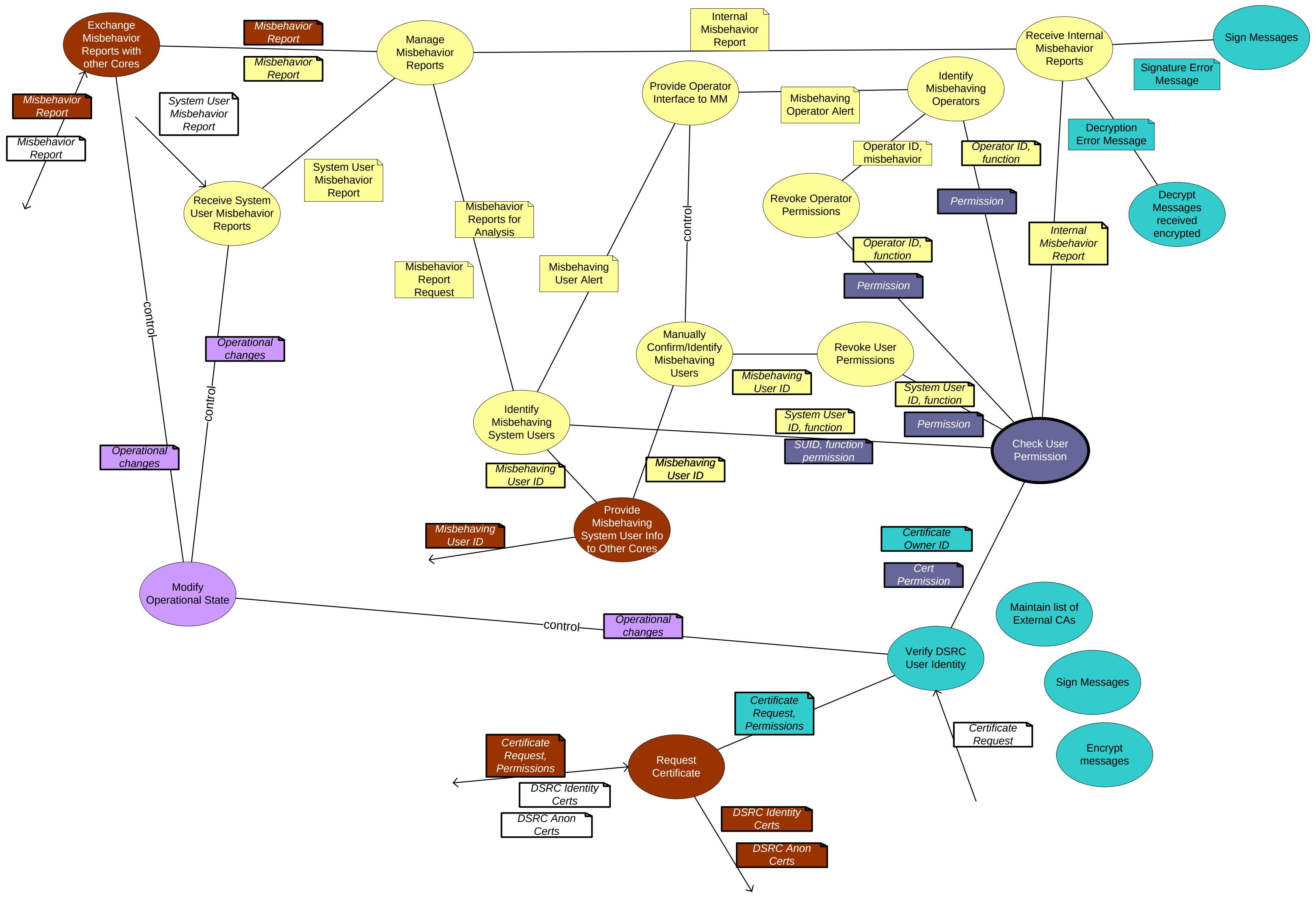

Exchange Misbehavior Reports with other Cores
Misbehavior Report
Misbehavior Report
Manage Misbehavior Reports
Internal Misbehavior Report
Receive Internal Misbehavior Reports
Sign Messages
Signature Error Message
Decryption Error Message
Decrypt Messages received encrypted
Misbehavior Report
Misbehavior Report
System User Misbehavior Report
Provide Operator Interface to MM
Misbehaving Operator Alert
Identify Misbehaving Operators
Operator ID, misbehavior
Operator ID, function
Receive System User Misbehavior Reports
System User Misbehavior Report
Misbehavior Reports for Analysis
Revoke Operator Permissions
Permission
Internal Misbehavior Report
Operator ID, function
Permission
Misbehavior Report Request
Misbehaving User Alert
control
control
Operational changes
control
Manually Confirm/Identify Misbehaving Users
Revoke User Permissions
Misbehaving User ID
System User ID, function
Identify Misbehaving System Users
System User ID, function
Permission
SUID, function permission
Check User Permission
Operational changes
Misbehaving User ID
Misbehaving User ID
Provide Misbehaving System User Info to Other Cores
Misbehaving User ID
Certificate Owner ID
Cert Permission
Modify Operational State
control
Operational changes
Maintain list of External CAs
Verify DSRC User Identity
Sign Messages
Certificate Request, Permissions
Certificate Request
Encrypt messages
Certificate Request, Permissions
Request Certificate
DSRC Identity Certs
DSRC Anon Certs
DSRC Identity Certs
DSRC Anon Certs

Figure 4-14: Functional View - DSRC Certificate Management Alternative 2

3) Both registration and certificate distribution could occur outside the Core. This removes much functionality including components of Misbehavior Management. It significantly increases the difficulty and impacts the flexibility of assignment of DSRC certificate permissions, because management of those permissions must stay with the CA. Thus local entities must establish a relationship with the CA-provider. The interface managing permissions at the Core becomes a human interface (contract, agreement) in an Enterprise View with no functional component inside the Core. In the Enterprise view this can be shown through the Core System Manager but it requires additional information, the permissions data, to be provided to the CA. This approach could co-exist with the approaches documented above but would be difficult to administer as ‘who runs what’ becomes complex (in some cases being managed by the Core, in other cases by an external or other-Core CA). Detection of misbehavior also becomes more complex with some Misbehavior Management occurring at each Core, but only the CA Core able to effectively act on it.

Figure 4-15 illustrates the Core that performs neither CA nor RA functions. Several functions from the basic architecture are deleted. Two additional objects are defined:

Provide Misbehaving System User Info to Other Cores: Provides the results of Misbehavior Management to other Cores. Misbehavior detection must still be handed by this Core because the RA is familiar with the user’s identity and roles and not the CA. This function provides the CA-core with the ID of misbehaving System Users and provides information necessary to revoke or modify their certificate.

Associated Information Objects:

- **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed, and is provided to other Cores that perform CA functions for further action.

Ask CA-Core to Revoke User Permissions: Provides the results of Misbehavior Management to other Cores. Misbehavior detection must still be handed by this Core for management of local permissions (e.g., who is allowed to use field applications managed by the Core operators). This function provides the CA-core with the ID of misbehaving System Users, and provides information necessary to revoke or modify their certificate.

Associated Information Objects:

- **System User ID, function** is a request sent to the CA-Core to revoke the System User’s permissions to carry out the indicated function.

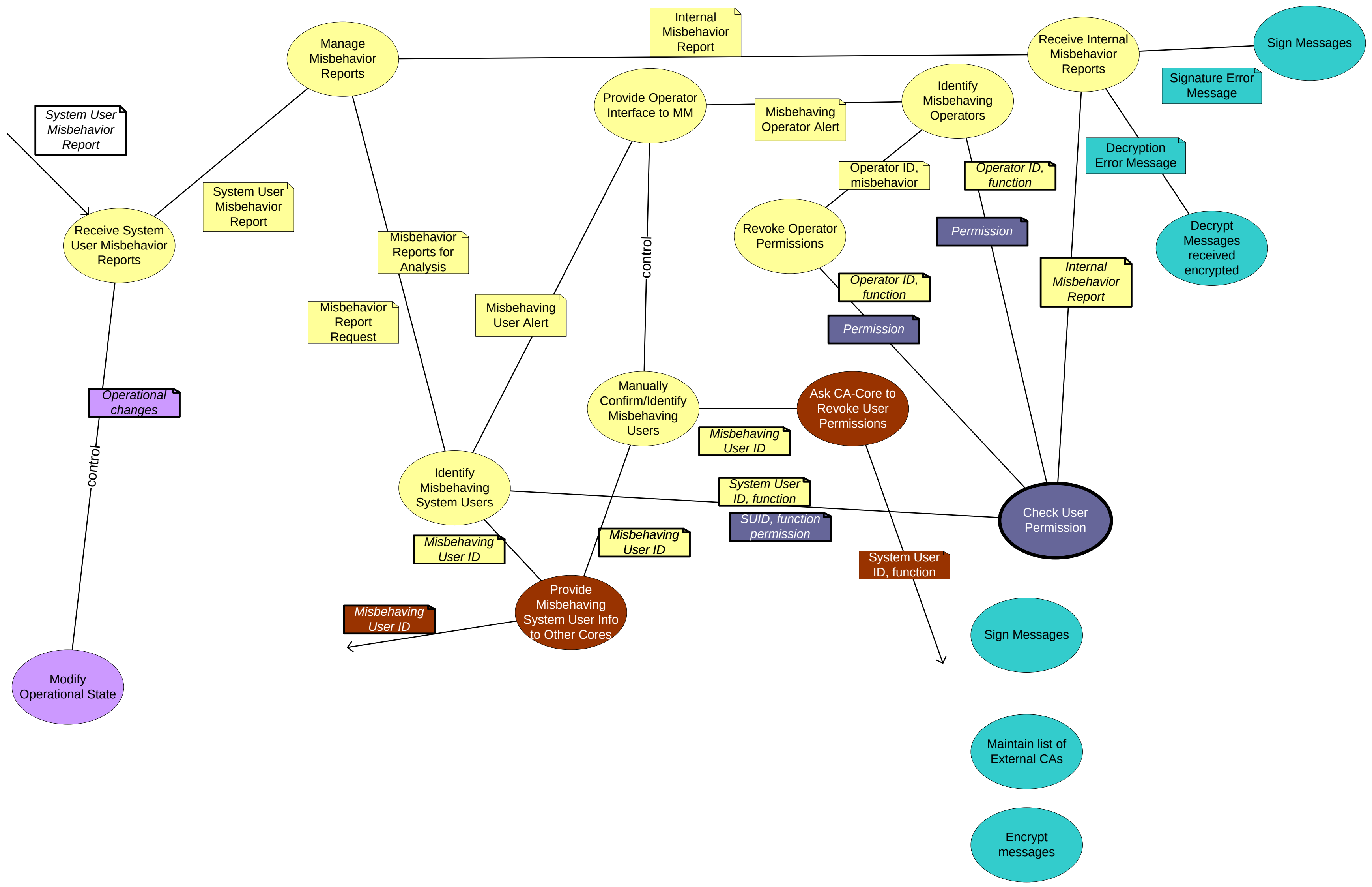

Manage Misbehavior Reports
Internal Misbehavior Report
Receive Internal Misbehavior Reports
Sign Messages
Signature Error Message
Provide Operator Interface to MM
Misbehaving Operator Alert
Identify Misbehaving Operators
System User Misbehavior Report
System User Misbehavior Report
Receive System User Misbehavior Reports
Operator ID, misbehavior
Operator ID, function
Decryption Error Message
Revoke Operator Permissions
Permission
Decrypt Messages received encrypted
Misbehavior Reports for Analysis
control
Operator ID, function
Internal Misbehavior Report
Misbehavior Report Request
Misbehaving User Alert
Permission
Operational changes
Manually Confirm/Identify Misbehaving Users
Ask CA-Core to Revoke User Permissions
Misbehaving User ID
control
Identify Misbehaving System Users
System User ID, function
Check User Permission
SUID, function permission
Misbehaving User ID
Misbehaving User ID
System User ID, function
Provide Misbehaving System User Info to Other Cores
Misbehaving User ID
Sign Messages
Modify Operational State
Maintain list of External CAs
Encrypt messages

Figure 4-15: Functional View - DSRC Certificate Management Alternative 3

4) DSRC anonymous certificate distribution could be moved outside the Core, but identity certificate distribution stay inside. This eliminates the difficulties associated with managing local permissions and enables a common CA for all anonymous System Users. This case could co-exist with the basic case, where one or more Cores could serve as CA and RA, but some Cores only CA for identity but only RA for anonymous users. No diagram is provided for this case, as it represents a removal of the basic cases' functionality but no additions.

5) DSRC anonymous users could be allowed to provide Misbehavior Reports. This requires only that Receive System User Misbehavior Reports be modified to allow such reports, and that System User Misbehavior Report allow a Pseudo-ID for reporter ID.

6) Each Core providing certificates could be configured as a root CA. This makes the job of coordinating certificates more difficult, and changes the security challenges. Since each Core is a root, if one Core is compromised then effectively all Cores are compromised until the vulnerability is discovered. (If only one Core serves as root, then the compromise of a sub-CA Core endangers System Users using that Core, but not necessarily other Cores or System Users using other Cores).

4.2.5 Functional View – X.509 Certificate Management

4.2.5.1 Introduction

The Core needs to provide X.509 certificates to Field and Center Users that connect directly to the Core, as well as Core internal systems that require certificates. Mobile Users connecting to the Core by 3G/4G are assumed to have their own X.509 certificate provided by their service provider.

4.2.5.2 Object Definitions and Roles

Check User Permission: This function accepts a System User ID or Operator ID, along with a type of operation that the user is attempting to access, and responds with whether or not the user is permitted that action. With regard to certificate distribution User Permissions maintains a listing of what each user is permitted to do; consequently it provides the information necessary to establish the permissions section of an X.509 certificate.

Associated Information Objects:

- **Permission** is the response from the Check User Permission function describing whether or not the **User** is permitted this action.
- **System User ID, function** is a request received from Identify Misbehaving System Users to determine whether the System User is permitted to carry out **function**.
- **SUID, Function Permission** is a response sent to Identify Misbehaving System Users indicating whether or not the System User is permitted to use the **function**.
- **Certificate Owner ID** is received from Provide X.509 Identity Certificates as a query to determine whether the System User is allowed to request an X.509 identity certificate.
- **Cert Permission** is the response sent to Provide X.509 Identity Certificates indicating whether the System User is allowed to request a X.509 identity certificate, and also the types of permissions that must be attached to the identity certificate.

Coordinate Certificate Distribution with Other Cores: This function allows the Core to share certificate distribution tasks with other Cores. This function distributes the IDs of the X.509 certificates that the Core has distributed with other Cores and receives the same from those other Cores.

Associated Information Objects:

- **X.509 Identity Certificate IDs** are the identifiers of X.509 identity certificates that the Core has distributed to System Users. This message is received from the Provide X.509 Identity Certificates function, and also passed on from other Cores to the Maintain X.509 Identity Certificates function.

Distribute X.509 CRL: This function distributes the X.509 Certificate Revocation List to System Users.

Associated Information Objects:

- **CRL** is the list of active X.509 certificates that are no longer valid.

Exchange CRLs with other Cores: Exchanges Certificate Revocation Lists (CRLs) with other Cores. This allows each Core to maintain a complete CRL so that System Users need receive a CRL from only one source.

Associated Information Objects:

- **Complete CRL** is a message containing the IDs of all certificates the sending Core knows to be invalid. This is sent by the Core or received from another Core.

- **CRL Deltas** is a message describing the changes to the CRL since the last CRL Delta or Complete CRL was sent to the Core that is receiving this message.

Identify Misbehaving System Users: Examines System User misbehavior reports, and identifies when users are malfunctioning or appearing to act maliciously.

Associated Information Objects:

- **Misbehavior Report Request** requests misbehavior reports from the Manage Misbehavior Reports function according to a set of criteria depending on the type of analysis to be done. The message may indicate reports that happened at a certain time interval, in a particular spatial area, and are attributable to a System User type or even specific System User.
- **Misbehavior Reports for Analysis** sent from the Manage Misbehavior Reports function contains the misbehavior reports requested by the Misbehavior Report Request.
- **Misbehaving User Alert** is sent to the Provide Operator Interface to MM function to inform Operators that a System User may be engaged in misbehavior. The message indicates the type of misbehavior and type and/or identity of the System User as applicable.
- **System User ID, function** is a request sent to Check User Permissions to determine whether the System User is permitted to carry out **function**.
- **SUID, Function Permission** is a response from Check User Permissions indicating whether or not the System User is permitted to use the **function**.

Maintain Core X.509 Certificate: The Core must obtain an X.509 certificate in order for it to operate a CA. This function accounts for obtaining that certificate from an external support system and managing that certificate.

Associated Information Objects:

- **Certificate Request:** The Core requests a certificate from an external CA.
- **X.509 Certificate**: The Certificate granted to the Core.
- **CRL**: The certificate revocation list provided periodically by the external CA.

Maintain X.509 Identity Certificates: This function maintains the inventory of X.509 identity certificates distributed by the Core including the key pairs associated with these certificates. It also maintains a list of the identifiers of identity certificates issued by other Cores. It provides these certificates upon valid request.

Associated Information Objects:

- **X.509 Identity Cert Request** is a request for a new identity certificate sent from the Provide X.509 Identity Certificates function on behalf of a System User.
- **X.509 Identity Certs** is a message containing one or more new identity certificates for the System User.
- **Activated X.509 Identity Cert IDs** contains the IDs of the X.509 identity certificates issued to a System User by other Cores, provided by the Coordinate Certificate Distribution with Other Cores function.

Manage X.509 CRL: Maintains the Certificate Revocation List for all X.509 certificates. It adds entries to the CRL based on inputs from Manually Confirm/Identify Misbehaving Users. It responds to queries about associations between System User IDs entries in the CRL.

Associated Information Objects:

- **CRL** is the complete certificate revocation list.
- **CRL Deltas** is a message containing changes to the CRL since the last time it was published.
- **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed and is received from the Manually Confirm/Identify Misbehaving Users function.
- **System User ID** is received from Provide X.509 Identity Certificates and includes the identity of a System User.
- **System User CRL Associations** is a response to the Provide X.509 Identity Certificates function that indicates the certificate IDs that are on the CRL and are associated with the System User ID previously provided.

Manually Confirm/Identify Misbehaving Users: Accepts input from the Provide Operation Interface to MM, and thus from the Operator, of a misbehaving System User and his misbehavior. This allows the Operator to manually identify misbehaving users.

Associated Information Objects:

- **Misbehaving User ID** contains the ID and type of misbehavior the System User has committed, and is provided to the Manage X.509 CRL and Revoke User Permissions functions for further action.

Modify Operational State: This is a controlling function that instructs various components to change the way they operate. In restricted mode the exchange of misbehavior reports and distribution of certificates will be constrained.

Associated Information Objects:

- **Operational Changes** describe how the X.509 certificate distribution functions need to alter operations. If in a fully operational state this will include restrictions on operations. If already restricted, this will be a change to restrictions, including possibly removing all restrictions and returning to fully operational mode.

Provide X.509 Identity Certificates: This function accepts requests for X.509 identity certificates from System Users and services valid requests by obtaining identity certificates from the Maintain X.509 Identity Certificates function and passing them along to the System User. It verifies the certificate request by examining the provided certificate and verifying its format and content, and then by querying Check User Permissions to verify that the owner of this certificate is allowed to request new certificates. If operating in a restricted mode, this function will prioritize certificate requests based on Core configuration settings (see the Maintain Core Operational Configuration function of Functional View – System Monitor and Control).

Associated Information Objects:

- **Certificate Request** is a request for a new identity certificate from a System User.
- **X.509 Identity Certs** is a message a new identity certificates for the System User.
- **X.509 Identity Cert Request** is a message to the Maintain X.509 Identity Certificates function requesting a new identity certificate on behalf of the System User.

- **X.509 Identity Cert IDs** contains the IDs of the X.509 identity certificates issued to a System User, provided to the Coordinate Certificate Distribution with Other Cores function.
- **Certificate Owner ID** is sent to Check User Permissions as a query to determine whether the System User is allowed to request a X.509 identity certificate.
- **Cert Permission** is the response from Check User Permissions indicating whether the System User is allowed to request a X.509 identity certificate, and also the types of permissions that must be attached to the identity certificates.
- **System User ID** is a query sent to Manage CRL (that includes the identity of a System User) asking for any certificate IDs associated with this user that are on the CRL.
- **System User CRL Associations** is a response from the Manage CRL function that indicates the certificate IDs that are on the CRL and are associated with the System User ID previously provided.

Provide Operator Interface to MM: Provides an interface to the Operator, allowing him access to Misbehavior Management functions. In this functional view is not concerned with control of operational states or modes, which is covered under a different view.

Associated Information Objects:

- **Misbehaving Operator Alert** is received from the Identify Misbehaving Operator function to inform other Operators that an Operator may be engaged in misbehavior.

Revoke User Permissions: Accepts n System User ID and an identification of misbehavior the System User is engaged in and subsequently revokes his permissions through the Check User Permissions function.

Associated Information Objects:

- **System User ID, misbehavior** is a message received from the Manually Confirm/Identify Misbehaving Users function. The message includes the System User ID, and an indication of the types of permissions to revoke.
- **System User ID, function** is a request sent to Check User Permissions to revoke the System User’s permissions to carry out the indicated function.
- **Permissions** is a response from Check User Permissions identifying the System User’s new permissions.

4.2.5.3 View Description

This view includes the functions necessary act as an X.509 Certificate Authority (i.e., distribution and management of DSRC certificates) and the detection of misbehavior associated with X.509 certificates. Within those broad high level functions:

- Distribution of X.509 certificates includes the distribution of both anonymous and identity certificates, including the generation of the public and private keys necessary to encrypt and decrypt messages and the maintenance and distribution of Certificate Revocation Lists (CRLs). It also includes the exchange of certificate distribution coordination information between Cores, and the ability to restrict certificate distribution when the Core changes operational state.

- Detection of misbehavior was largely documented in Functional View – DSRC Certificate Distribution. Misbehavior-related functions that directly interface with X.509 certificate management functions are shown on this view to identify those interfaces.

- Both certificate distribution and misbehavior management require coordination between Cores, to ensure that CRLs are consistent, that no two Cores issue the same certificates, and to improve misbehavior detection by sharing misbehavior reports between Cores. For X.509 certificate distribution, Core’s will serve as sub-CA to an external root CA provided by a third party.

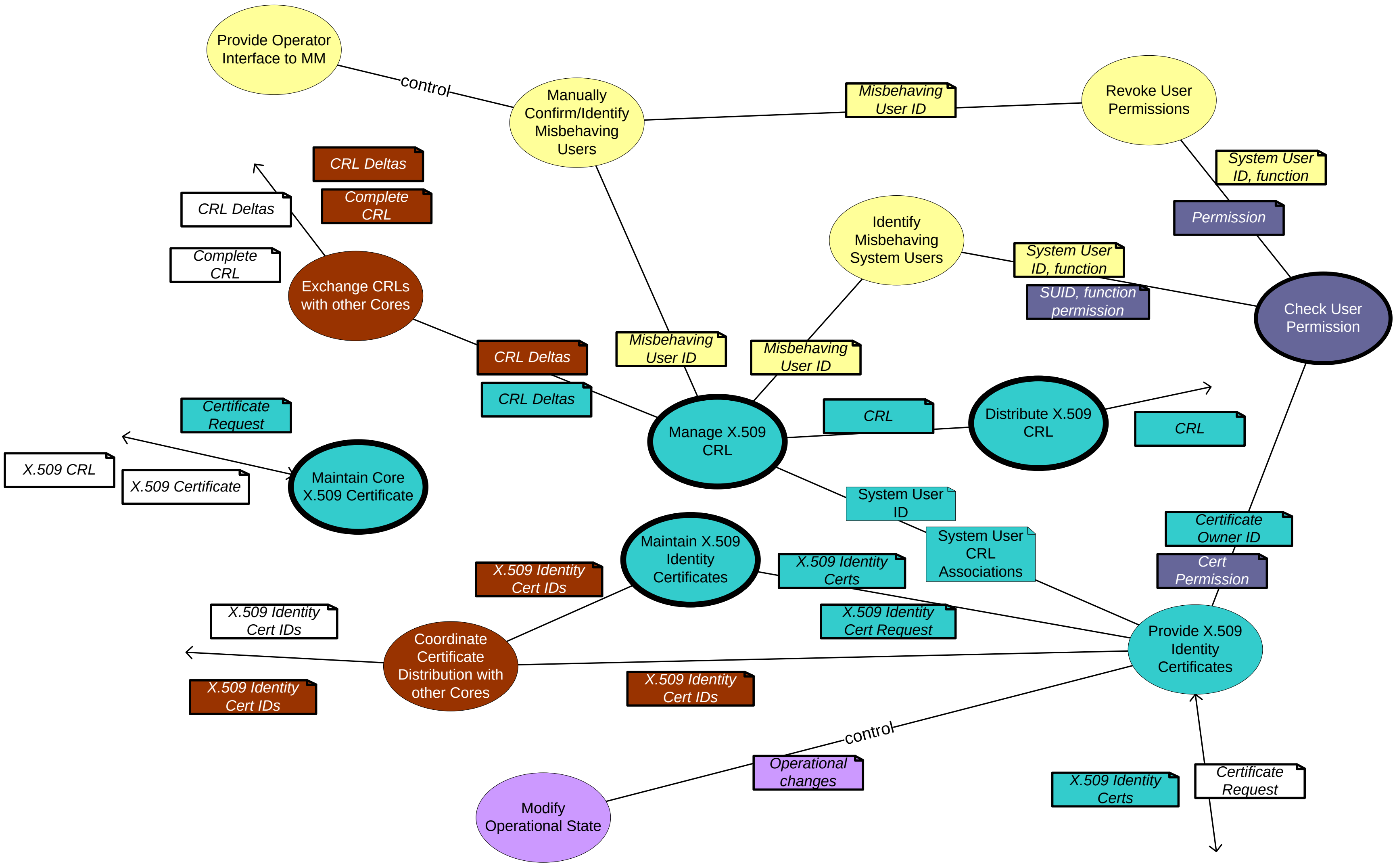


Figure 4-16: Functional View – X.509 Certificate Management

4.2.5.4 Configuration Information

The following views must be considered when changing this view:

Enterprise Viewpoint: Enterprise View – Governance
Functional Viewpoint: Functional View – DSRC Certificate Distribution
Connectivity Viewpoint: Connectivity View – High Level

4.2.5.5 Alternatives Considered

1) The Core System could serve as a root CA. This may require more trust from third parties, since all X.509 certificate providers would have to agree to trust the Core. With large numbers of Cores operating this way, the web of trust could grow quite large. This creates more security concerns, as compromising one Core would endanger all X.509 users. This approach is not recommended.
2) A single Core System could serve as root CA, effectively replacing the external 3rd party root CA described above. This could work well with the Enterprise View that includes a Core Certifying Body; the Root Core could work with that body and not issue a Core any certificates until it had passed certification. Unfortunately, this puts one Core in a dominating position. Given that Cores could be operated by competitive corporations, this architecture is not market equitable.

4.2.6 Functional View – Core Decryption

4.2.6.1 Introduction

Encrypted messages meant for the Core must be decrypted in order for the Core to act on their contents. Decryption using PKI schemes requires the storage of a private key at the receiver. If the Core is to be scalable, it must be able to operate across multiple nodes (see the Connectivity View) which implies the storage of that same private key in multiple places. This makes security more difficult, as there would be multiple points that, if compromised, could surrender the private key. This view documents a mechanism that mitigates this problem.

4.2.6.2 Object Definitions and Roles

Application Component: The Application Component function is a representative function, representing any other function requiring decryption of a remotely received encrypted message.

Associated Information Objects:

- **Remotely Encrypted Message** is an encrypted message intended for the Application Component.
- **Decrypted Message** is the decrypted form of the Remotely Encrypted Message.

Decrypt Message Received Encrypted: This function has access to the Core's secret key. It uses that key to decrypt messages and provides that decrypted message to the Encrypt Message Using Core Local Key function.

Associated Information Objects:

- **Decrypted Message** is the decrypted version of the Remotely Encrypted Message.

Encrypt Message Using Core Local Key: Encrypts the now-decrypted message with a local encryption mechanism known to all Core subsystems.

Associated Information Objects:

- **Decrypted Message** is the decrypted form of the Remotely Encrypted Message.
- **Locally Encrypted Message** is the locally encrypted form of the decrypted Remotely Encrypted Message.

Decrypt Locally Encrypted Message: Decrypts the locally encrypted form of the Remotely Encrypted Message.

Associated Information Objects:

- **Locally Encrypted Message** is the locally encrypted form of the decrypted Remotely Encrypted Message.
- **Decrypted Message** is the decrypted form of the Remotely Encrypted Message.

Maintain Core Secret Key: This function stores the Core's secret key.

Associated Information Objects:

- **Secret Key** is the Core's secret key.

Notify Misbehavior of Failed Decryption: In the event that a message cannot be decrypted even though it appeared to be intended for the Core, this event will be forwarded to Misbehavior Management.

Associated Information Objects:

- **Decryption Error Message** describes the decryption failure and includes a copy of the encrypted message. This message is provided by the Decrypt Message Received Encrypted function and is forwarded to the Receive Internal Misbehavior Reports function.

Provide Encrypted Message to Decryptor: Provides the Remotely Encrypted Message to the Decrypt Message Received Encrypted function.

Associated Information Objects:

- **Remotely Encrypted Message** is an encrypted message intended for the Application Component.

Receive Internal Misbehavior Reports: Collects misbehavior reports from Core subsystems.

Associated Information Objects:

- **Decryption Error Message** describes the decryption failure and includes a copy of the encrypted message. This message is received from the Notify Misbehavior of Failed Decryption function.

4.2.6.3 View Description

The view below shows the functional decomposition of the special object Decrypt Message. This decomposition identifies two components dedicated solving the decryption problem, and three functions related to the Core function application that requires the message to be decrypted.

When an Application Component of any service has received a message addressed to the Core System that it needs to have decrypted with the Core's private key it forwards that message to a message passing component. This message passing component is separated from the Application Component to decouple the application from its local communications processing. The message passing component (Provide Encrypted Message to Decryptor) passes the message to the Decrypt Messages received encrypted function. This function has access to the private key. While not shown on this view, the connectivity view makes clear that this function exists on only one node, thus requiring only one location to store this key.

The decrypted message is then passed to the encryption function, which encrypts the message using an encryption algorithm known by the Decrypt Locally Encrypted Message component. For example this could be a symmetric algorithm using pre-shared keys. Once encrypted, the message is passed to a receiver that can decrypt the locally encrypted message and pass the now-decrypted message to the Application Component that needs its content.

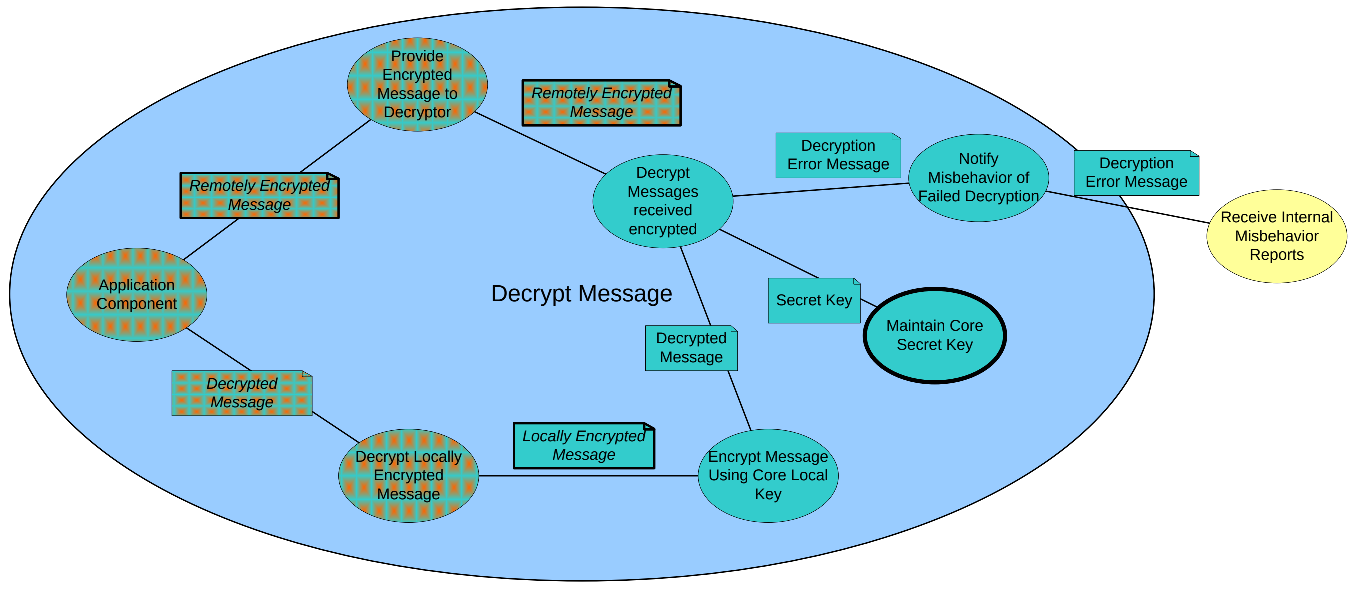

Provide Encrypted Message to Decryptor
Remotely Encrypted Message
Decryption Error Message
Notify Misbehavior of Failed Decryption
Decryption Error Message
Remotely Encrypted Message
Decrypt Messages received encrypted
Receive Internal Misbehavior Reports
Application Component
Decrypt Message
Secret Key
Maintain Core Secret Key
Decrypted Message
Decrypted Message
Locally Encrypted Message
Decrypt Locally Encrypted Message
Encrypt Message Using Core Local Key

Figure 4-17: Functional View - Core Decryption

4.2.6.4 Configuration Information

The following views must be considered when changing this view:

Connectivity Viewpoint: Connectivity View – High Level
Communications View: Communications View – Mobile DSRC Device and Core

4.2.6.5 Alternatives Considered

The Core System's private encryption key could be stored at each node that required the ability to read encrypted messages directed to the Core. The Core's performance could still scale, particularly if each subsystem with an external interface had a different private key, but storage of private keys at multiple locations would increase the exposure of information (the key) that could be used to compromise the Core. This alternative was rejected to reduce this exposure.

Selecting this alternative forces the selection of alternatives for the views noted under Configuration Information.

4.2.7 Functional View – Networking

4.2.7.1 Introduction

In order to provide services to System Users the Core must be connected to the Internet. This view explores the functionality required to maintain security and provide communications for the Core.

4.2.7.2 Object Definitions and Roles

<u>**Special Information Objects:**</u>
All Data In/Out represents any message sent to or from a function. Messages may be encrypted and/or signed.

Intrusion Detection: Identifies and reports malicious network and system activity.
Associated Information Objects:

- **Intrusion Alert** is sent to the Service Monitor's Provide Operator Interface function and documents an identified malicious action

Intrusion Prevention: Identifies, reports, and blocks malicious activity. Data that is not known to be associated with malicious activity is forwarded to the Receive Data/Request function.
Associated Information Objects:

- **Intrusion Alert** is sent to the Service Monitor's Provide Operator Interface function, and documents an identified malicious action and/or the status of Intrusion Prevention's response to the action.

Generic Application Component: The Generic Application Component function is a representative function, representing any other Functional Object.

Monitor Service Control Node Performance: This function monitors the performance of application components on Service Control Nodes, and provides summary information to Route Data/Request.
Associated Information Objects:

- **Service Control Node Performance** describes the performance and loading of application components operating on Service Control Nodes.

Provide Internet Connectivity: Provides the Internet Connectivity, including routing functions.

Route Data/Request: This function receives data, which may be a request for service, data intended for data distribution other data intended for the Core, and routes the data to the appropriate Service Control Node application component. It performs this routing based on the destination required for the data, and the performance data it receives concerning Service Control Node performance. This function balances application performance given available resources.
Associated Information Objects:

- **Service Control Node Performance** describes the performance and loading of application components operating on Service Control Nodes.

4.2.7.3 View Description

The view below shows the basic functionality required of the Service Router Node. All data intended for the Core passes through this node. In addition to providing Internet connectivity, routing and firewall services, the Service Router monitors traffic for malicious activity and distributes messages to the application component best able to handle the traffic given current performance.

One of the disadvantages of this approach is that all network traffic must be passed through the Intrusion Prevention System. This creates a bottleneck, as the system's maximum capacity is limited by the throughput of this function. It also introduces additional latency into the data stream, though no Core functions have such low latency requirements that this will have a significant effect.

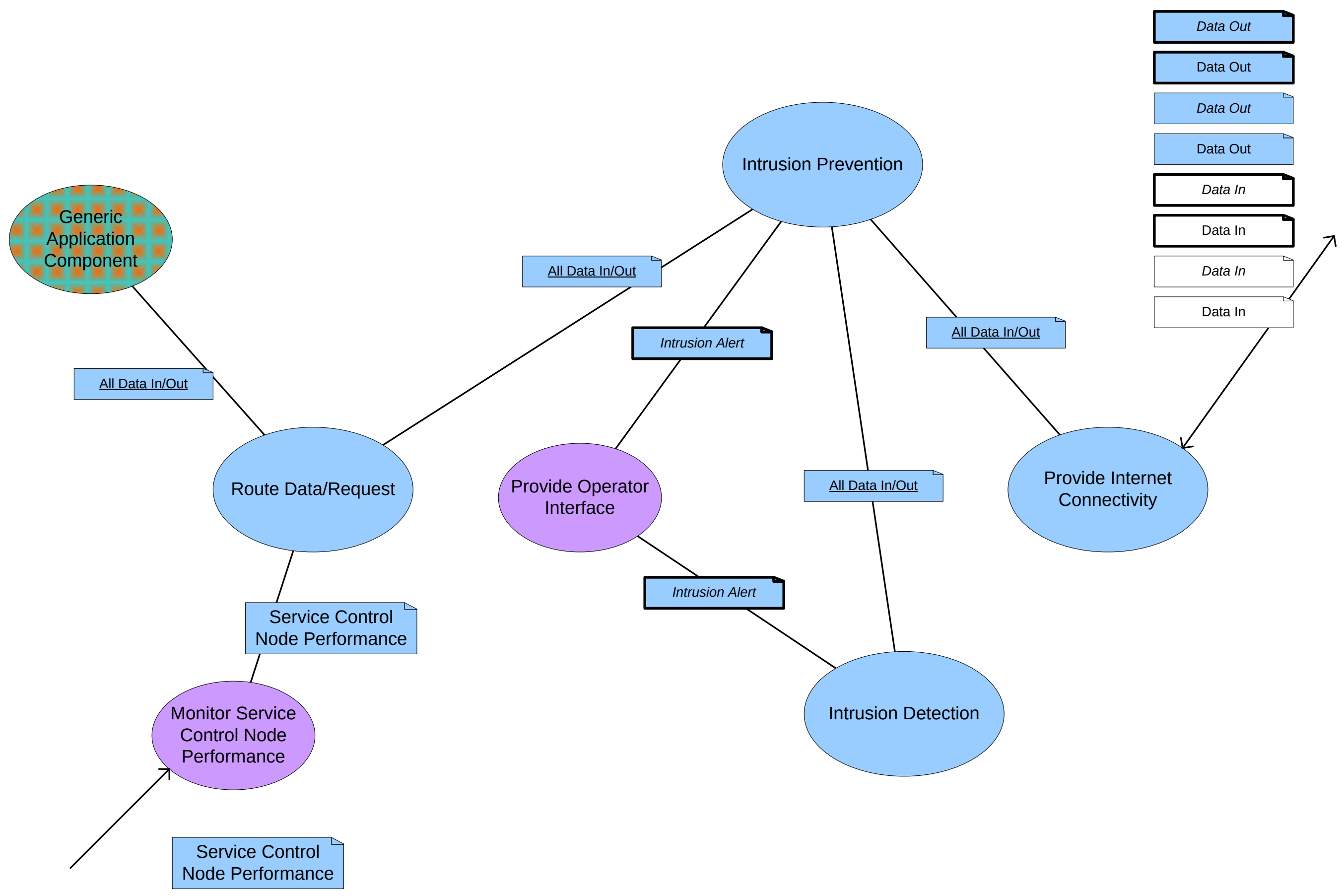


Figure 4-18: Functional View - Network Connectivity

4.2.7.4 Configuration Information

The following views must be considered when changing this view:

Connectivity Viewpoint: Connectivity View – High Level

4.2.7.5 Alternatives Considered

The alternative illustrated in Figure 4-19 removes the Intrusion Prevention function. This removes a bottleneck to performance and reduces implementation cost, but makes the Core more vulnerable to cyber-attack.

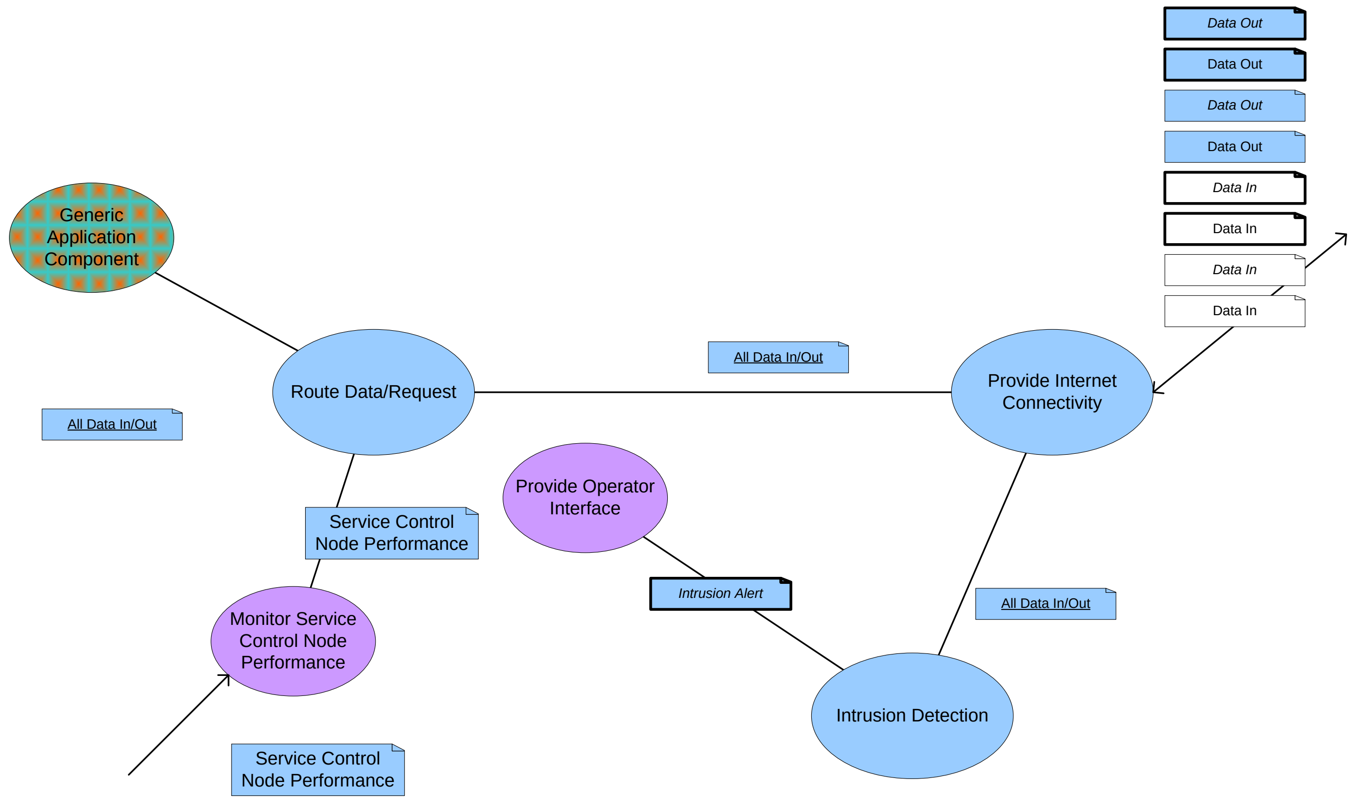

Data Out
Data Out
Data Out
Data Out
Data In
Data In
Data In
Data In
Generic Application Component
All Data In/Out
Route Data/Request
All Data In/Out
Provide Internet Connectivity
Provide Operator Interface
Service Control Node Performance
Monitor Service Control Node Performance
Intrusion Alert
All Data In/Out
Intrusion Detection
Service Control Node Performance

Figure 4-19: Functional View - Network Configuration Alternative no IPS

4.2.8 Functional View – Core Backup

4.2.8.1 Introduction

Core Systems may provide backup functionality to one another. This view addresses the functionality required to implement Core backup.

4.2.8.2 Object Definitions and Roles

Object definitions and roles will be defined in a future version of this document. Relevant requirements from the SyRS that will be satisfied by this view:

System Requirements:

The Core2Core Subsystem shall provide a backup capability with interfacing Core Systems.

Subsystem Requirements:

The Core2Core Subsystem shall provide backup services to an interfacing degraded Core System.

The Core2Core Subsystem shall send data stores for facility backup to preselected interfacing Core Systems.

The Core2Core Subsystem shall accept requests to operate as facility backup to preselected interfacing Core Systems.

The Core2Core Subsystem shall send requests to operate as facility backup to preselected interfacing Core Systems

The Core2Core Subsystem shall provide facility backup of a variable subset of Core System services in the case of unavailability of to an interfacing Core System

The Core2Core Subsystem shall provide backup services to an interfacing failed Core System.

4.2.8.3 View Description

TBD

4.2.8.4 Configuration Information

The following views must be considered when changing this view:

TBD

4.2.8.5 Alternatives Considered

TBD

4.3 Connectivity Viewpoint

Connectivity Viewpoint

Connections between Nodes (hardware), Links (interfaces) and Applications (software) (OSI 7)

4.3.1 Connectivity View – High Level

4.3.1.1 Introduction

In order to promote scalability the Core needs to have its functionality deployed across multiple nodes. This view shows the highest level view of the distributed Core and how it is connected to System Users.

4.3.1.2 Object Definitions and Roles

Core Nodes: All Core Nodes are located in a fixed, environmentally controlled, and secured facility. They are connected to one another by wired communications.

Core Decryptor: This node is responsible for providing decryption of encrypted messages intended for the Core, and providing those messages back to the Core function that requires them, in a secured form that the function can interpret.

Service Component Node: The service provider for the Core System. Service Component nodes provide Core functionality. They route all traffic to System Users through the Service Router and receive communications from System Users through the Service Router.

Service Router: The gateway for the Core System, the Service Router provides proxies for all Core services and forwards requests for service to Service Component Nodes.

User Nodes: End Users of Core services.

Center User Node: This is the Center User. This user is fixed in space and must be connected to the Internet.
Field User Node: This is the Field User. This user is fixed in space and must be connected to the Internet.
Mobile User Node: This is the Mobile User, connecting wirelessly by 5.9 GHz DSRC to the DSRC Field Node or through cellular or Wi-Fi communications to the Cellular Node.

Communications Nodes: These nodes provide communications between User Nodes and Core Nodes.

Internet Node: This is the publically accessible Internet, through which all users must communicate to gain access to the Core.
Cellular Node: This represents the cellular infrastructure, including 3G and 4G infrastructure, which provides a gateway to Core services through the Internet.
DSRC Field Node: This represents 5.9 GHz DSRC-based roadside equipment (RSE) that provide a gateway to Core services through the Internet.

4.3.1.3 View Description

The Core includes three main physical component nodes: the Service Router, Service Component Node and Core Decryptor.

The Service Router provides the interface between the Internet and Service Component Nodes. All communications traffic between the Core and a System User go through the Service Router. The Service Router provides proxies for all services operated by the Core, and selects a Service Component Node to receive traffic depending on load.

A Core has a minimum of one Service Component Node but no maximum. Service Component Nodes run software processes that provide Core functionality. Nodes may be organized by subsystem (e.g. User Permissions node) or function (e.g. Maintain DSRC Anonymous Certificates). This allows the Core to scale.

A Core has a single Core Decryptor node. This node includes the functionality for decrypting messages intended for the Core, and re-encrypting them using Core-specific security methods.

Mobile User Nodes are connected wirelessly. All other nodes can be connected wirelessly or by wire.

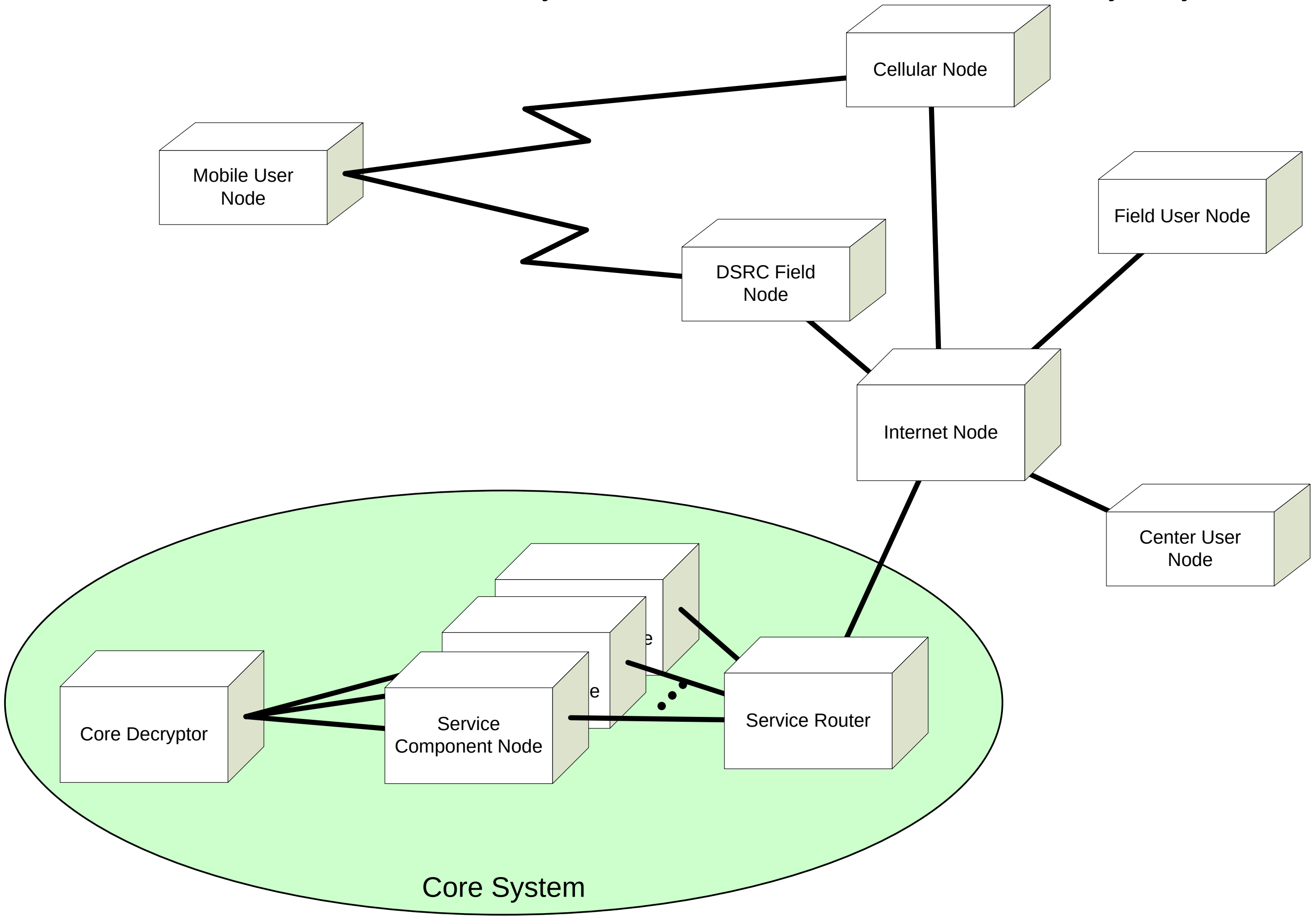


Figure 4-20: Connectivity View High Level

4.3.1.4 Configuration Information

The following views must be considered when changing this view:

Functional Viewpoint: Functional View – Core Decryption
Communications View – Mobile DSRC Device and Core

4.3.1.5 Alternatives Considered

1) Mobile, Field and/or Center Users could connect to the Core by a private network, instead of being required to use the Internet. This option is illustrated in Figure 4-21. It is compatible with the base architecture. If allowed, Mobile, Field and Center Users could connect to the Core by either Internet or Private Network.

   Private Network Node: This is a privately controlled-network that has a relationship with Mobile, Center and or Field Users as well as the Core. Traffic between those System Users and the Core traverses this private network.

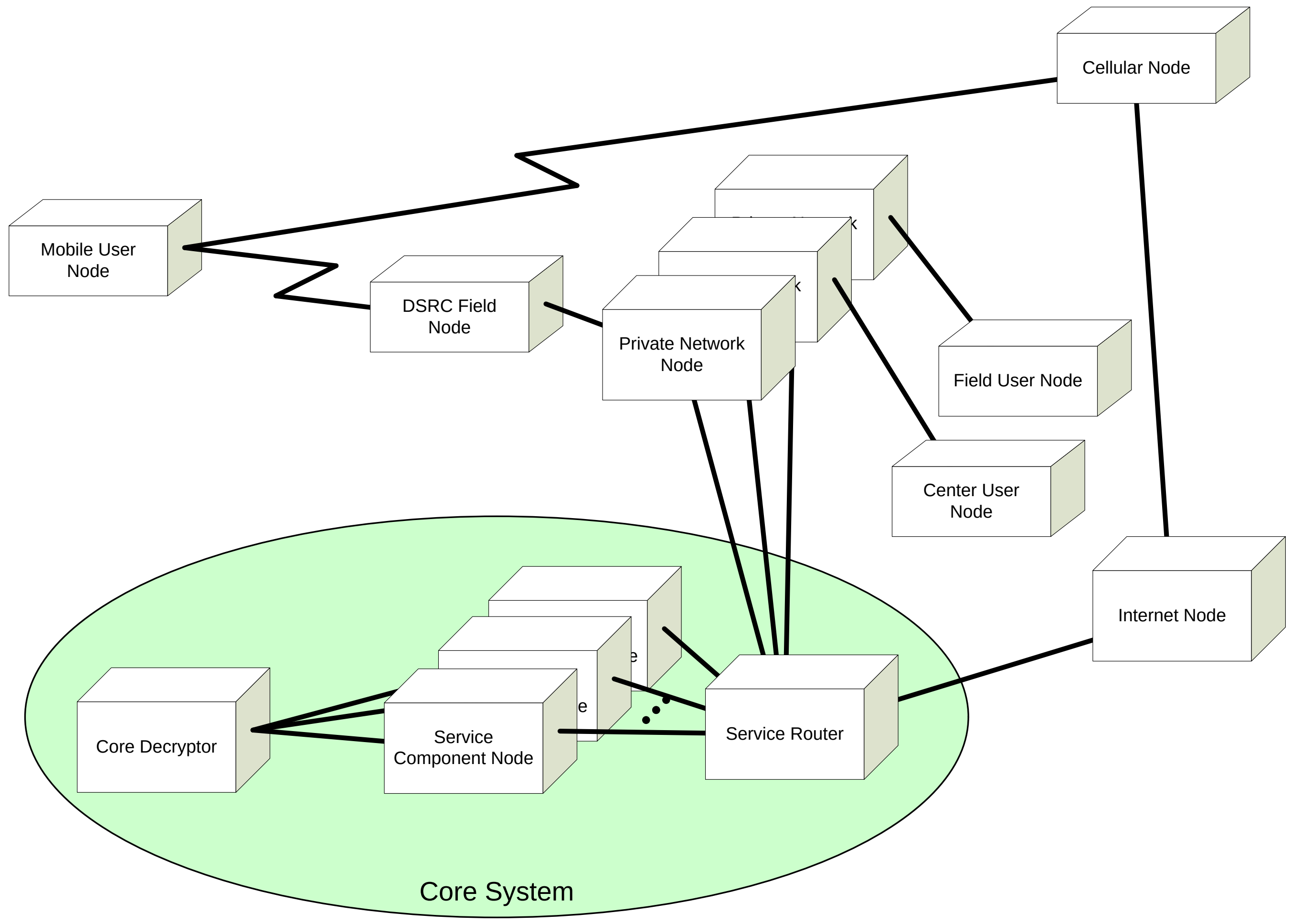


Figure 4-21: Connectivity View High Level Alternative

2) The Core Decryptor Node is not necessary if the alternative view for Functional View – Core Decryption is adopted. Private keys would be stored on multiple Service Component Nodes.

4.3.2 Connectivity View – Core System Function Allocation

4.3.2.1 Introduction

This view illustrates the allocation of Core System Functional Objects to component nodes. Most Functional Objects are treated similarly; those related to decryption, local encryption, service routing, and intrusion detection have specific allocations as illustrated.

4.3.2.2 Object Definitions and Roles

Core Decryptor: This node is responsible for providing decryption of encrypted messages intended for the Core, and providing those messages back to the Core function that requires them, in a secured form that the function can interpret.

The Core Decryptor is physically connected to all Service Component Nodes. Any message requiring decryption is passed from a Service Component Node to the Core Decryptor which performs the decryption and returns the message in locally encrypted form.

Functional and Information objects allocated to the Core Decryptor are documented in Functional View – Core Decryption.

Service Component Node: The service provider for the Core System. Service Component nodes provide Core functionality. They route all traffic to System Users through the Service Router and receive communications from System Users through the Service Router.

Service Component Nodes receive data from System Users through the Internet and the Service Router. All data (be it signed and/or encrypted, no matter its purpose) reaches Service Component Nodes only after passing through the Service Router. Similarly all data sent to a System User is first sent to the Service Router.

Functional and Information objects allocated to the Core Decryptor are documented in Functional View – Core Decryption.

Service Router: The gateway for the Core System, the Service Router provides proxies for all Core services and forwards requests for service to Service Component Nodes.

Functional and Information objects allocated to the Core Decryptor are documented in Functional View – Networking.

4.3.2.3 View Description

Multiple Service Component Nodes are not shown for clarity. Network switches that may be necessary to connect Service Component Nodes, Core Decryptor and Service Router are not shown either; the choice of Core internal network technology and topology is left to the implementer.

Encrypted messages may be forwarded directly from the Service Router to the Core Decryptor, or if they are included as part of a larger message that could be addressed to an Application Component, only passed to the Core Decryptor by the Application Component when it requires the message to be decrypted.

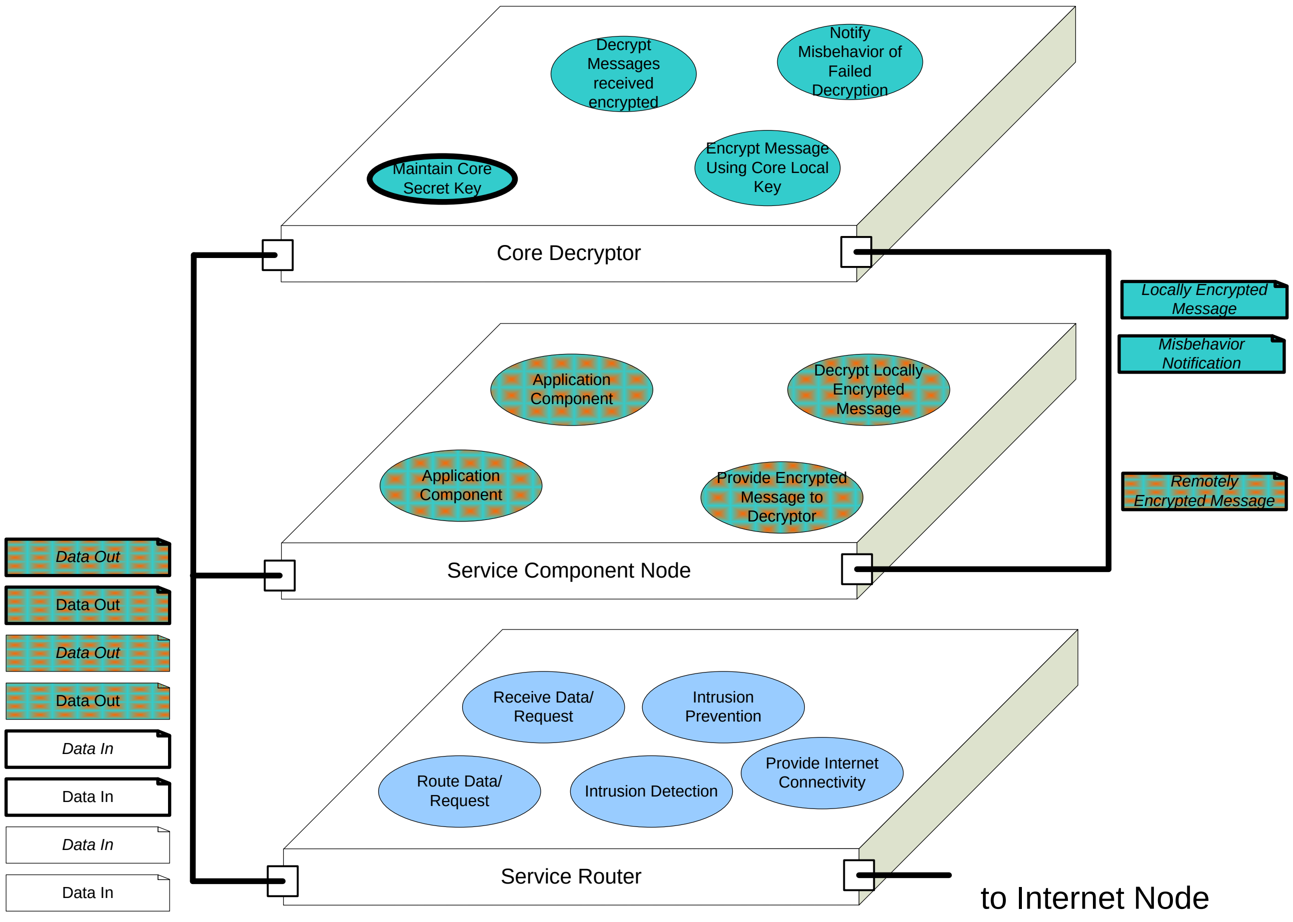


Figure 4-22: Connectivity View - Node Functional Allocation

4.3.2.4 Configuration Information

The following views must be considered when making changes to this view:

Functional Viewpoint: Functional View – Data Distribution
Functional Viewpoint: Functional View – Core Decryption
Functional Viewpoint: Functional View – Networking
Connectivity Viewpoint: Connectivity View – High Level
Communications Viewpoint: Communications View – Mobile DSRC Device and Core.

4.3.2.5 Alternatives Considered

As shown in Figure 4-23, this alternative shows the Core Decryptor only accessible by Service Component Nodes. Any encrypted message must either have a non-encrypted portion indicating its destination function or there must be a new function created to serve as unaddressed encrypted message proxy on the Service Component Node; this function would receive such messages and pass them to the Core Decryptor, who would have decryption forward the messages to the appropriate Service Component Node. This mirrors one of the alternatives in the Communications Viewpoint: Communications View – Mobile DSRC Device and Core.

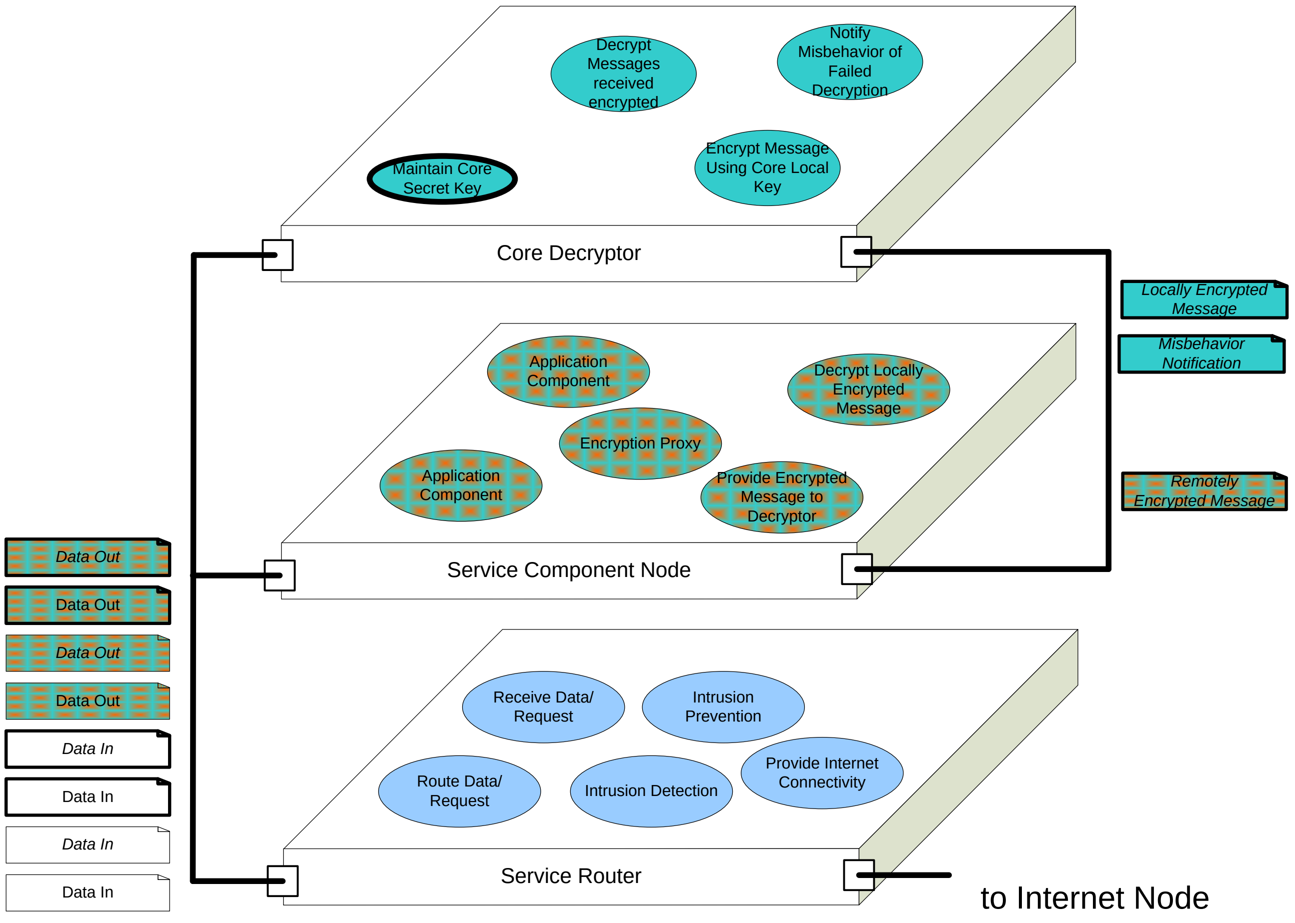


Figure 4-23: Connectivity View - Node Functional Allocation Alternative

4.4 Communications Viewpoint

Communications Viewpoint
Layered communications protocols (OSI 1-5) between Nodes (hardware)

The Communications View defined here is one of many several to come. A future version of this document will include Mobile Cellular to Core communications, Field and Center to Core communications, and Core-to-Core communications.

4.4.1 Communications View – Mobile DSRC Device and Core

4.4.1.1 Introduction

In order for Mobile Users to communicate with the Core System, they must use some form of wireless communications. This is the first of two views exploring Mobile-Core communications; this view focuses on Mobile Users using DSRC. Since DSRC communication is by definition short range, an intermediary must accept the DSRC message and provide it to the Core.

4.4.1.2 Object Definitions and Roles

Mobile DSRC Device: The Mobile User node equipped with a DSRC radio

> **Application Mobile Component:** This is the Functional Object that wants to communicate with the Core, most likely software.
> **DTLS:** Datagram Transport Layer Security
> **Internet Protocols:** UDP and IPv6
> **Low Level Protocols:** IEEE 1609.4, IEEE 802.11p

The Mobile DSRC Device and DSRC Field Node communicate across a wireless medium.

DSRC Field Node: The field node providing DSRC connectivity to the Mobile User with backhaul capable of accessing the Core System.

> **DTLS:** Datagram Transport Layer Security
> **Internet Protocols:** UDP/TCP and IPv6
> **Low Level Protocols:** IEEE 1609.4, IEEE 802.11p
> **DTLS/TLS:** Datagram Transport Layer Security or Transport Layer Security, depending on whether the link between Field Node and the Core System uses UDP or TCP, respectively.
> **Low Level WAN:** A variety of wide area network communications media are possible, including 3G/4G cellular, WiMAX, T1/T3

Service Router: Service Router is the Core System node that receives all System User communications and routes data to the appropriate Core function. Unencrypted data received from the Field Node is passed to the appropriate Service Component Node. If the data received from the Field Node is encrypted, the data is forwarded to the Decryption Node. Re-encrypted data received from the Decryption Node is passed to the appropriate Service Component Node.

> **Internet Protocols:** UDP/TCP and IPv6
> **LTLS:** Locally implemented Transport Layer Security. This is TLS, but between Core nodes.
> **Low Level LAN:** A variety of local area network communications media are possible, but most likely switched Ethernet at 1 Gbps.

Service Component Node: The Core System node that provides service functionality. Any Functional Object other than those assigned to decryption or service routing operates on a Service Component Node.

Decryption Node: The Core System node that provides decryption of messages intended for the core, re-encrypts them using local encryption keys, and sends the message data back to the Service Router.

Internet Protocols: UDP/TCP and IPv6
DTLS/TLS: Datagram Transport Layer Security or Transport Layer Security depending on whether the link between Field Node and the Core System uses UDP or TCP, respectively.
LTLS: Locally implemented Transport Layer Security. This is TLS but between Core nodes.
Low Level LAN: A variety of local area network communications media are possible, but most likely switched Ethernet at 1 Gbps.

4.4.1.3 View Description

Communications from a Mobile User using DSRC start with a Mobile User – Field Node interaction. If the communication is encrypted, the DTLS layer of the protocol stack will be used, otherwise it will be bypassed (this is not shown on the diagram for clarity).

The Field Node communicates with the Core System using IP protocols. If the Mobile User communication was encrypted the field node will use DTLS or TLS to maintain a similar level of security. The Field Node is not as latency-concerned when communicating with the Core, so it could use TCP and thus TLS instead of UDP and DTLS.

A message that arrives at the Core System's Service Router will be passed to the relevant Functional Object on an available Service Component Node. If the message is encrypted it may need to first be passed to the Core Decryption Node, decrypted, re-encrypted and then returned to the Service Router, where it will then be passed to the appropriate Service Component Node.

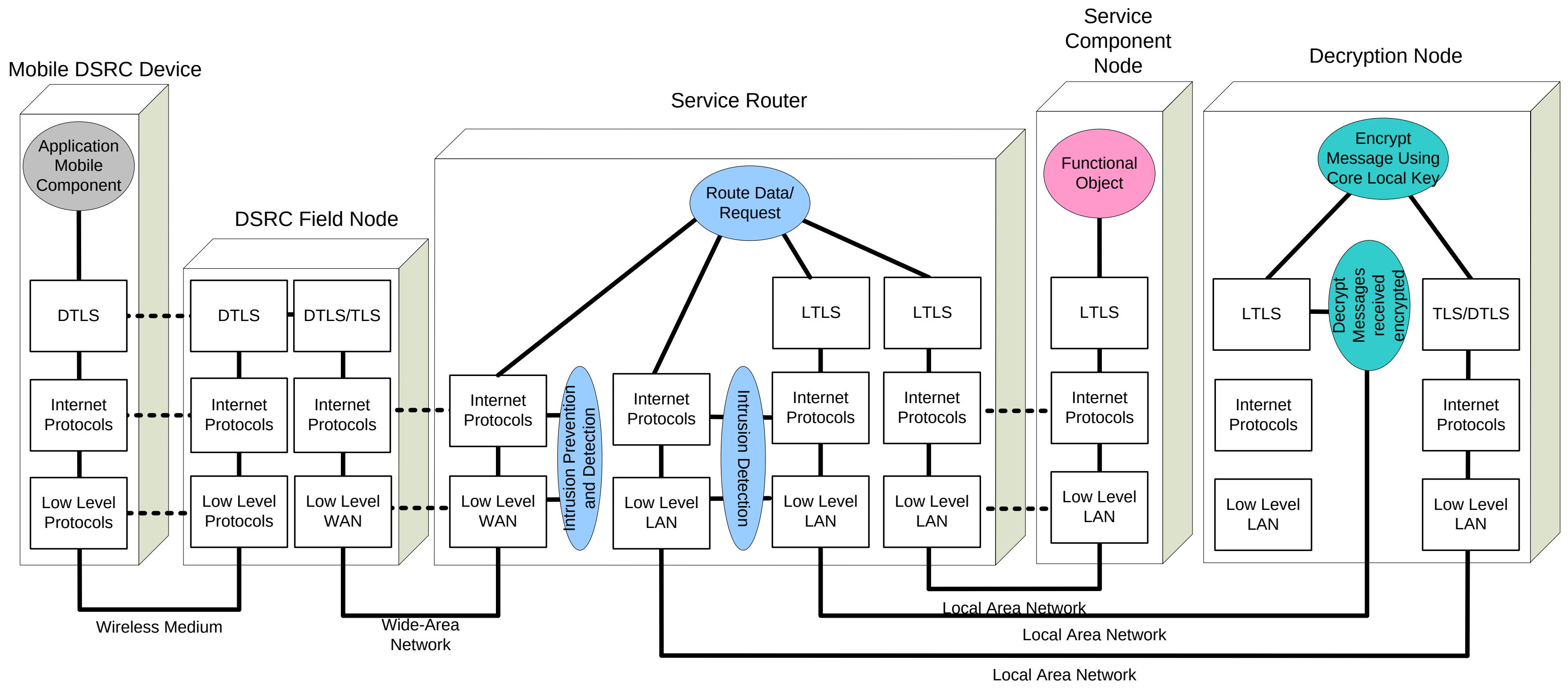


Figure 4-24: Communications View - DSRC Mobile to Core

4.4.1.4 Configuration Information

The following views must be considered when making changes to this view:

Connectivity Viewpoint: Connectivity View – High Level
Connectivity Viewpoint: Connectivity View – Core System Function Allocation
Functional View – Core Decryption

4.4.1.5 Alternatives Considered

1) If the message payload were encrypted, but not the Core System address which includes where to route the service request, the linkage between Core Decryptor and Service Router would be replaced by a linkage between Service Component Node and Core Decryptor.
2) If the alternative Core Decryption and High Level Connectivity Views are selected where the Core's private key is distributed among multiple nodes, then this communications view get significantly simpler. The Decryption Node goes away, with decryption functionality residing on each Service Component Node. This removes the need for local TLS.

4.4.2 Communications View – Mobile Cellular User and Core

4.4.2.1 Introduction

TBD.

4.4.2.2 Object Definitions and Roles

TBD.

4.4.2.3 View Description

TBD.

.

4.4.2.4 Configuration Information

TBD.

4.4.2.5 Alternatives Considered

TBD.

4.4.3 Communications View – Center/Field User and Core

4.4.3.1 Introduction

TBD.

4.4.3.2 Object Definitions and Roles

TBD.

4.4.3.3 View Description

TBD.

4.4.3.4 Configuration Information

TBD.

4.4.3.5 Alternatives Considered

TBD.

4.4.4 Communications View – Core and Core

4.4.4.1 Introduction

TBD.

4.4.4.2 Object Definitions and Roles

TBD.

4.4.4.3 View Description

TBD.

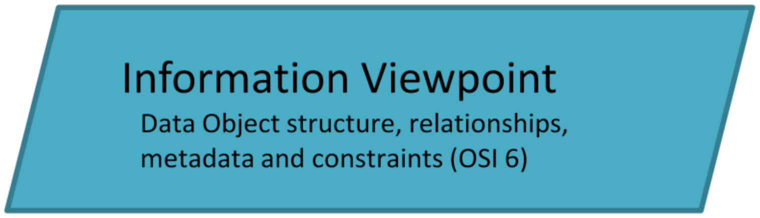
4.4.4.4 Configuration Information

TBD.

4.4.4.5 Alternatives Considered

TBD.

4.5 Information Viewpoint


Information Viewpoint
Data Object structure, relationships, metadata and constraints (OSI 6)

The Information View defined here is one of many several to come. While much work has gone into the definition of message objects passed to and from vehicle-based DSRC-using Mobile Users (see the SAE J2735 standard), the Core System will have to define the messages that pass across the interfaces it provides to all System Users, and also the messages various Core System Software Engineering Objects exchange with one another.

Top Level Functional Objects are defined in various Functional Views. At the highest level shown here (the first two Information Views), Information Objects are exchanged by Functional Objects. Subsequent work beyond the scope of this SRD will have to define messages that are passed between the implementations of those Functional Objects (i.e., Software Engineering Objects)

4.5.1 Information View – Top Level External Objects

4.5.1.1 Introduction

This view describes the objects that are exchanged, accepted and sent over the Core System's external interfaces. This top level view does not describe individual messages. The objects defined in this view can be further decomposed to define messages that are exchanged between the Cores and System Users.

4.5.1.2 Object Definitions

All objects in this view are defined in Functional View – Top Level.

4.5.1.3 View Description

External Information Objects are organized according to the subsystem that they originate from or terminate in.

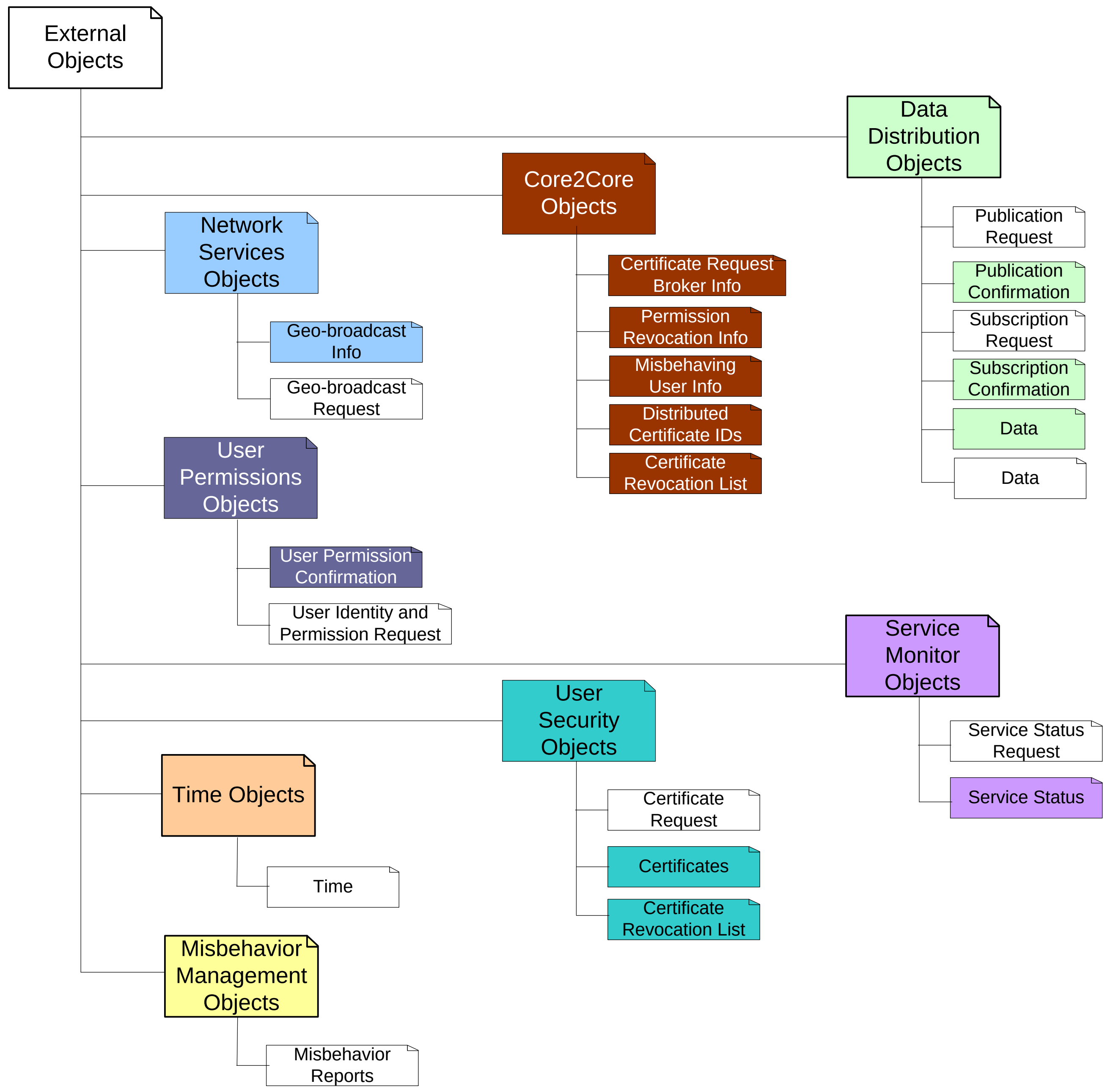


Figure 4-25: Information View - Top Level External Objects

4.5.1.4 Configuration Information

The following views must be considered when making changes to this view:

Connectivity Viewpoint: Connectivity View – High Level

4.5.1.5 Alternatives Considered

None.

4.5.2 Information View – Top Level Internal Objects

4.5.2.1 Introduction

This view describes the objects that are exchanged, accepted and sent over the Core System's internal interfaces. This top level view does not describe individual messages. The objects defined in this view can be further decomposed to define messages that are exchanged between the Core Software Engineering Objects.

4.5.2.2 View Description

TBD.

4.5.2.3 Configuration Information

TBD.

5.0 CONSISTENCY AMONG ARCHITECTURAL VIEWS

This section will be expanded later to provide an analysis of consistency across all of the architectural views and to record any inconsistencies among the architectural views.

One key consideration in an architecting process concerns view consistency. IEEE 1471 requires that each view be consistent. But IEEE 1471 allows a conforming architectural description to have two views, each of which is complete, to be inconsistent with each other. This would allow the existence of some software components in a functional view that are not allocated to a hardware component in a connectivity view. Clearly cross-view consistency is a goal, but this is often too difficult to achieve in practice. Instead, the requirement is to record known inconsistencies and provide an analysis of the inconsistencies.

6.0 ARCHITECTURAL RATIONALE

This section will be expanded later to present trade-offs considered, alternatives not chosen, or other analyses that led to choosing the architecture described in this System Architecture Document.

7.0 APPENDICES

Table 7-1: Viewpoint Summary

Specifications	Enterprise	Functional	Connectivity	Communication	Information
Overview	Relationship between organizations	Logical interactions between functional objects	Connections between nods (hardware) links (interfaces) applications (software)	Layered communications protocols between nodes	Data object structure, relationship meta data constraints
Defined Terms	• Enterprise objects • Facilities • Domain • Federation	• Functional Objects • Realized Information Objects	• Engineering Objects • Node • Link • Port • Application	• Protocol Entity	• Information Objects • Metadata • Information Package
Stakeholders	• Users (Mobile, Field , Center) • Acquirer • Operator • Maintainer • Developer • Manager • Policy-Setter	• Users (Mobile, Field , Center) • Acquirer • Operator • Maintainer • Developer • Manager • Tester	• Acquirer • Users (Mobile, Field , Center) • Maintainer • Developer • Tester	• Users (Mobile, Field , Center) • Maintainer • Developer • Manager • Tester • Policy-Setter	• Users (Mobile, Field , Center) • Operator • Developer • Tester
Concerns	• Security • Organization/Resources • Risks • Evolvability • Deployability • Maintainability	• Performance • Interfaces • Functionality • Security • Appropriateness • Feasibility • Evolvability • Maintainability	• Performance • Interfaces • Functionality • Security • Feasibility • Evolvability • Deployability	• Performance • Interfaces • Functionality • Security • Organization/Resources • Feasibility • Risks • Deployability	• Security • Applicability
Method(s) to Model/Represent Conforming Views	Table 3 2: Enterprise View Graphical Object Definitions	Table 3-3: Functional View Graphical Object Definitions	Table 3-4: Connectivity View Graphical Object Definitions	Table 3-5: Communications View Graphical Object Definitions	Table 3-6: Information View diagrams Graphical Objects Definition
Viewpoint Source	CCSDS 311.0-M-1	CCSDS 311.0-M-1	CCSDS 311.0-M-1	CCSDS 311.0-M-1	CCSDS 311.0-M-1
Security Issues	• Organizational Roles • Policies • Trust Relationships • Domain Boundaries • Cross-support Security Agreements	Access control interfaces on functions and specific functional elements	Physical elements that are used to implement security policies and barriers	Implementation of security protocols between: • Mobile Users and the Core System, • Mobile Users and other System Users	Protection from unauthorized access

8.0 GLOSSARY AND ACRONYMS

Term	**Definition**
Access Control	Refers to mechanisms and policies that restrict access to computer resources. An access control list (ACL) for example, specifies what operations different users can perform on specific files and directories.
Analysis	The process of studying a system by partitioning the system into parts (functions, components, or objects) and determining how the parts relate to each other.
Anonymity	Lacking individuality, distinction, and recognizability within message exchanges.
Application	A computer software program with an interface enabling people to use the computer as a tool to accomplish a specific task.
Authentication	The process of determining the identity of a user that is attempting to access a network.
Authorization	The process of determining what types of activities or access are permitted on a network. Usually used in the context of authentication: once you have authenticated a user, they may be authorized to have access to a specific service.
Assumption	A judgment about unknown factors and the future which is made in analyzing alternative courses of action.
Back Office	See Center
Bad Actor	A role played by a user or another system that provides false or misleading data, operates in such a fashion as to impede other users, operates outside of its authorized scope.
Center	An entity that provides application, management, administrative, and support functions from a fixed location not in proximity to the road network. The terms "back office" and "center" are used interchangeably. Center is a traditionally a transportation-focused term, evoking management centers to support transportation needs, while back office generally refers to commercial applications. From the perspective of the Core System ConOps these are considered the same.
Concept of Operations (ConOps)	A user-oriented document that describes a system's operational characteristics from the end user's viewpoint.
Constraint	An externally imposed limitation on system requirements, design, implementation or on the process used to develop or modify a system. A constraint is a factor that lies outside – but has a direct impact on – a system design effort. Constraints may relate to laws and regulations or technological, socio-political, financial, or operational factors.

Term	**Definition**
Contract	In project management, a legally binding document agreed upon by the customer and the hardware or software developer or supplier; includes the technical, organizational, cost, and/or scheduling requirements of a project.
Data Consumer	A user or system that is receiving or using data from another user or system.
Data Provider	A user or system that is supplying or transmitting data to another user or system.
Deployability	Able to be deployed in existing roadway environments, without requiring replacement of existing systems in order to provide measurable improvements.
Digital Certificates	A digital certificate is an electronic "identification card" that establishes your credentials when doing business or other transactions on the Web. It is issued by a certification authority. It contains your name, a serial number, expiration dates, a copy of the certificate holder's public key (used for encrypting messages and digital signatures), and the digital signature of the certificate-issuing authority so that a recipient can verify that the certificate is real. Note: From the SysAdmin, Audit, Network, Security Institute - www.sans.org website.
Encryption	Scrambling data in such a way that it can only be unscrambled through the application of the correct cryptographic key.
End-User	The ultimate user of a product or service, especially of a computer system, application, or network.
Environment	The circumstances, objects, and conditions that surround a system to be built; includes technical, political, commercial, cultural, organizational, and physical influences as well as standards and policies that govern what a system must do or how it will do it.
Extensibility	The ability to add or modify functionality or features with little or no design changes.
Flexibility	The ability to adjust or adapt to external changes with little or no design changes.
Functionality	The capabilities of the various computational, user interface, input, output, data management, and other features provided by a product.
Geo-cast	The delivery of a message to a group of network destinations identified by their geographical locations.
Hardware	Hardware refers to the physical parts of a computer and related devices. Internal hardware devices include motherboards, hard drives, and memory. External hardware devices include monitors, keyboards, mice, printers, and scanners.
Jurisdictional Scope	The power, right, or authority to interpret and apply the law within the limits or territory which authority may be exercised.
Maintainability	To keep in an existing operational state preserved from failure or decline of services (with minimum repair, efficiency, or validity).

Term	Definition
Permission	Authorization granted to do something. To the Core System, permissions are granted to System Users and Operators determining what actions they are allowed to take when interacting with the Core.
Priority	A rank order of status, activities, or tasks. Priority is particularly important when resources are limited.
Privacy	The ability of an individual to seclude information about themselves, and thereby reveal information about themselves selectively.
Problem domain	A set of similar problems that occur in an environment and lend themselves to common solutions.
Pseudo-ID	An identifier used by an individual that is not associated with the individual's identity, and may change to prevent such association.
Reliability	Providing consistent and dependable system output or results.
Scalability	The capable of being easily grown, expanded or upgraded upon demand without requiring a redesign.
Service	A set of related functionalities accessed using a prescribed interface.
Software	Software is a general term that describes computer programs. Terms such as software programs, applications, scripts, and instruction sets all fall under the category of computer software.
Special Permissions	Authorization granted to perform actions specific to 3rd party applications, using IEEE 1609.2 certificates as the permission grant mechanism. Also called certificate-managed application permissions.
States or Modes	A distinct system setting in which the same user input will produce different results than it would in other settings. The Core System as a whole is always in one state. Individual subsystems may be in different modes.
System	(A) A collection of interacting components organized to accomplish a specified function or set of functions within a specified environment. (B) A group of people, objects, and procedures constituted to achieve defined objectives of some operational role by performing specified functions. A complete system includes all of the associated equipment, facilities, material, computer programs, firmware, technical documentation, services, and personnel required for operations and support to the degree necessary for self-sufficient use in its intended environment.

Term	Definition
Traceability	The identification and documentation of derivation paths (upward) and allocation or flow down paths (downward) of work products in the work product hierarchy. Important kinds of traceability include: to or from external sources to or from system requirements; to or from system requirements to or from lowest level requirements; to or from requirements to or from design; to or from design to or from implementation; to or from implementation to test; and to or from requirements to test.
User	An individual who uses a computer, program, network, and related services of a hardware and/or software system, usually associated with granting that individual with an account and permissions.
User-ID	An identifier associated with a specific individual, that can be traced to that individual's identity.

Table 8-1: Acronyms and Abbreviations used in this Document

Abbreviation/ Acronym	Definition
CA	Certificate Authority
CAMP	Crash Avoidance Metrics Partnership
ConOps	Concept of Operations
COTS	Commercial off-the-shelf
CRL	Certification Revocation List
DSRC	Dedicated Short Range Communication
DMS	Dynamic Message Sign
DTLS	Datagram Transport Layer Security
FHWA	Federal Highway Administration
FTA	Federal Transit Administration
Gbps	Gigabits per second
IEEE	Institute for Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
ITS	Intelligent Transportation Systems
JPO	Joint Program Office
NHTSA	National Highway Traffic Safety Administration
PII	Personally Identifiable Information
RA	Registration Authority
RITA	Research and Innovative Technology Administration

Abbreviation/ Acronym	**Definition**
RSE	Roadside Equipment
SAD	System Architecture Document
SAE	Society of Automobile Engineers
SyRS	System Requirements Specification
SysML	Systems Modeling Language
TLS	Transport Layer Security
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
USDOT	US Department of Transportation
VII	Vehicle Infrastructure Integration
WAVE	Wireless Access in Vehicular Environments
WiMAX	Worldwide Interoperability for Microwave Access